2010

Semantic Provenance: Modeling, Querying, and Application in Scientific Discovery

Satya Sanket Sahoo
Wright State University

Semantic Provenance: Modeling, Querying, and Application in Scientific Discovery

A dissertation submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Computer Science and Engineering

By

SATYA SANKET SAHOO
Masters in Computer Application, Goa University, 2003

2010
Wright State University

WRIGHT STATE UNIVERSITY
SCHOOL OF GRADUATE STUDIES

July 29, 2010

I HEREBY RECOMMEND THAT THE DISSERTATION PREPARED UNDER MY SUPERVISION BY Satya Sanket Sahoo ENTITLED Semantic Provenance: Modeling, Querying, and Application in Scientific Discovery BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF Doctor of Philosophy in Computer Science and Engineering.

Amit P. Sheth, Ph.D.
Dissertation Director

Arthur Ardeshir Goshtasby, Ph.D.
Director, Computer Science and
Engineering Ph.D. Program

Andrew T. Hsu, Ph.D
Dean, School of Graduate Studies

Committee on
Final Examination

Krishnaprasad Thirunarayan, Ph.D.

Michael Raymer, Ph.D.

Nicholas V. Reo, Ph.D.

Olivier Bodenreider, Ph.D.

William S. York, Ph.D.

ABSTRACT

Sahoo, Satya Sanket. Ph.D., Department of Computer Science and Engineering, Wright State University, 2010.
Semantic Provenance: Modeling, Querying, and Application in Scientific Discovery

Provenance metadata, describing the history or lineage of an entity, is essential for ensuring data quality, correctness of process execution, and computing trust values. Traditionally, provenance management issues have been dealt with in the context of workflow or relational database systems. However, existing provenance systems are inadequate to address the requirements of an emerging set of applications in the new eScience or Cyberinfrastructure paradigm and the Semantic Web. Provenance in these applications incorporates complex domain semantics on a large scale with a variety of uses, including accurate interpretation by software agents, trustworthy data integration, reproducibility, attribution for commercial or legal applications, and trust computation. In this dissertation, we introduce the notion of "*semantic provenance*" to address these requirements for eScience and Semantic Web applications.

In addition, we describe a framework for management of semantic provenance by addressing the three issues of, (a) provenance representation, (b) query & analysis, and (c) scalable implementation. First, we introduce a foundational model of provenance called Provenir to serve as an upper-level reference ontology to facilitate provenance interoperability. Second, we define a classification scheme for provenance queries based on the query characteristics and use this scheme to define a set of specialized provenance query operators. Third, we describe the implementation of a highly scalable query engine to support the provenance query operators, which uses a new class of materialized views based on the Provenir ontology, called Materialized Provenance Views (MPV), for query optimization.

We also define a novel provenance tracking approach called Provenance Context Entity (PaCE) for the Resource Description Framework (RDF) model used in Semantic Web applications. PaCE, defined in terms of the Provenir ontology, is an effective and scalable approach for RDF provenance tracking in comparison to the currently used RDF reification vocabulary. Finally, we describe the application of the semantic provenance framework in biomedical and oceanography research projects.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank the team members of the Bioinformatics for Glycan Expression (glycomics) project - at the University of Georgia: Will York, James Atwood, Lin Lin, and John Miller; and at the Kno.e.sis Center (Wright State University): Cory Henson, and Christopher Thomas. The glycomics project was a very exciting real world research context for being introduced to biomedical ontologies, large scale data integration, scientific workflows, and provenance.

I would also like to thank the team members of the Semantic Problem Solving Environment for *Trypanosoma Cruzi* project – at the University of Georgia: Brent Weatherly, Todd Mining, Rick Tarleton, and Prashant Doshi; at the Kno.e.sis Center: Vinh Nguyen, and Priti Parikh; and other members of the Kno.e.sis Center: Pascal Hitzler, Guozhu Dong, Pramod Anantraman, Harshal Patni, Ajith Ranabahu, Prateek Jain, Delroy Cameron, Pablo Mendes, Hemant Purohit, Wenbo Wang, Ramakanth Kavaluru, Ashwin Manjunatha, Meena Nagarajan, Pavan Kapanipati, Ashutosh Jadav, and Raghava Mutharaju with whom I have enjoyed having stimulating academic and social conversations over the past several years.

Working with Olivier Bodenreider, at the Lister Hill Center (National Institutes of Health), over the past four years has been a source of immense joy that has not only been translated to satisfying research results but has been a tremendous learning experience in working with biomedical research challenges. I was fortunate to have had the opportunity to work with Roger Barga, at Microsoft Research (MSR), who provided the unique insight of bridging the gap between traditional data management and provenance research. Interactions with Roger and Jonathan Goldstein (MSR) played a critical role in formulating and implementing many components of the provenance management framework presented in this thesis.

Thanks especially to Will York for guiding me through the challenges of cutting edge life sciences research and helping me acquire the essential skills for inter-disciplinary research. Thanks also to Krishnaprasad Thirunarayan for helping me gain an understanding of logic languages and for working with me to formalize the various components of this thesis. Thanks to Michael Raymer and Nick Reo for serving on my PhD committee.

Thanks to Sushmita Sheshadri for patiently explaining the fundamentals of cell biology. Special thanks to Boanerges Aleman-Meza for being an amazing role model and for always willing to help.

Most importantly, I am indebted to my advisor, Amit Sheth, who has shaped my research philosophy, work ethics, and career objectives. With his ability to “think big”, relentless focus on quality, and giving unprecedented opportunities to his students to expand their research horizon, it has been a fascinating experience for me to explore and learn. I am deeply influenced by his energy, his dedication to the success of his students, and sincerely hope to advise any future students working with me in a similar manner.

To my parents, Maheswar and Bijanbala, and sister, Sonal

Chapter 1

Introduction

The phenomenal growth in computing capabilities is transforming scientific research from being an experiment-driven discipline to a "data-driven" science [1]. Data driven science is enabled by the synergistic use of a new generation of scientific instruments along with the exponentially growing World Wide Web (WWW) infrastructure. This infrastructure, also called eScience or Cyberinfrastructure [2], enables scientists to generate and access resources, such as "raw" experiment data, online structured databases, repositories of textual data, and Web-based computing resources (Web services). The move towards eScience has created a deluge of scientific data [2] that are exemplified by the datasets available at the Cancer Biomedical Informatics Grid (caBIG) [3] and Biomedical Informatics Research Network (BIRN) [4] [5]. This paradigm shift places a greater emphasis on the processing, integration, and analysis of data, with significant impact on scientific research, as compared to the earlier focus on the generation of data.

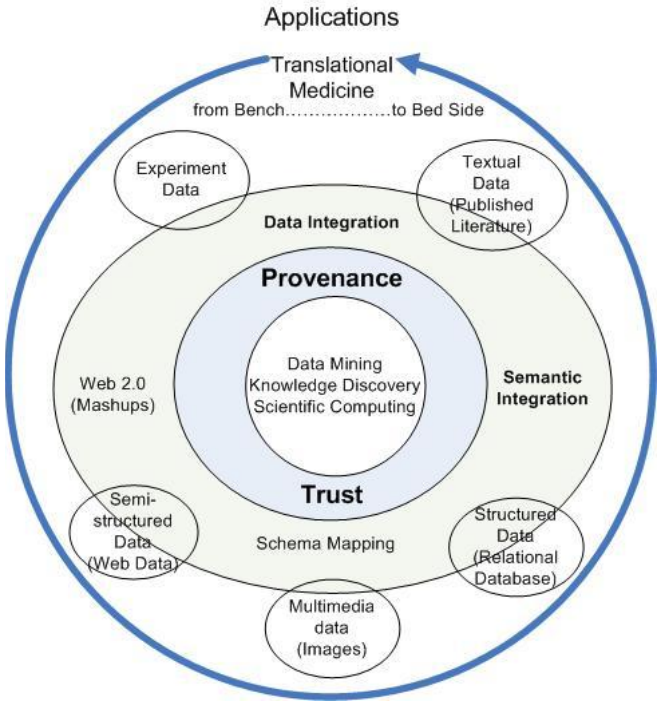


Figure 1-1: Overview of "data-driven science" research

Scientists require a new generation of tools and applications to leverage the large amount of data to not only guide ongoing projects, but also help drive future research. The size of scientific data presents a

set of unique challenges to the database, data mining and Semantic Web research communities to create scalable solutions for analysis, scientific visualization, question answering, and knowledge discovery tasks (Figure 1-1).

The role of metadata to address the challenge of managing large scale data and supporting user services with interactive response time has been the subject of research in a variety of domains including multimedia [6], relational database, sensor networks, and eScience. Metadata, a term coined by Jack Myers in 1969 [7], is information that describes the attributes of data, for example size of a file, the schema of a relational database, spatial and temporal information associated with a sensor reading. Metadata's critical role in processing and analyzing large volumes of data has long been understood in library management [8], geography [9], multimedia [10], the biological sciences [11], and the Web [12].

1.1 Metadata: Syntax, Structure, and Semantics

The primary advantage of metadata is the level of abstraction it provides as compared to the data itself, which allows data management applications to use metadata for a variety of tasks including data selection, processing, visualization, integration, sharing, and retrieval. The database community, in particular, has extensively explored the use of metadata to exchange, share, and integrate data from heterogeneous information sources [13] [14]. There are many types of metadata including annotations, content-specific metadata, security specification, processing instruction, usage information, data quality, and descriptions [15] [16]. There is a large body of work on classification of metadata [17], [18], [19], for example the classification scheme proposed in [15] categorizes metadata into two broad categories of:

1. **Content independent metadata**: This class of metadata does not depend on the content of associated dataset. For example, the type of sensor or the location of a file
2. **Content dependent metadata**: This class of metadata depends on the content of the associated dataset and can be further subdivided into
    a) Direct content based metadata (for example a document vector for full-text indices)
    b) Content descriptive metadata (for example, the encoding used to create a multimedia file)

The research in data management (representation, integration, processing, and querying) has evolved from a purely syntactical approach to structural and then to the use of semantics to effectively reflect domain-specific complexities [20]. A similar approach has been argued for metadata management that resulted in the definition of semantic metadata. Semantic metadata is "metadata that describes contextually relevant or domain-specific information about content (optionally) based on a[n]… ontology" [21]. Semantic metadata not only mitigates terminological heterogeneity but also enables software applications to "understand" (through explicit and precise description of terms) and reason over it. Specific motivating factors for using semantic metadata are:

1. To create a conceptual context to "capture domain knowledge and help impose a conceptual semantic view on the underlying data"
2. For accurate data interpretation
3. To support metadata interoperability

The use of semantic metadata on the WWW [10] an important precursor to the introduction of the Semantic Web initiative [22].

1.2 Semantic Web

In contrast to the traditional content on the WWW that is meant for human consumption, the Semantic Web initiative, led by the World Wide Web consortium (W3C), envisions the creation of a "Web of data" [23] that is not only accessible to humans but is also "understood" by software agents through use of semantic metadata. The Semantic Web impacts a vast range of applications and tasks including large scale data integration, enhanced Web search, support for complex domain-specific queries, data visualization, and knowledge discovery tasks. The Semantic Web is underpinned by a number of recommended standards, with a pivotal role played by the Resource Description Framework (RDF) [24]. It is a representation model consisting of three elementary components of Subject, Predicate, and Object (S,P,O) that can be used to describe any entity and its attributes on the Web [24]. An example of an RDF statement is "`protein→encoded_by→gene`" that states that a gene encodes a protein, where protein is the S, the gene is the O, and S and O are linked by a named relationship or P `encoded_by`. The RDF model, along with the RDF Schema (RDFS) [25] and Web Ontology Language (OWL) [26], have formal semantics that enables software applications such as reasoners to process, interpret, and discover implicit knowledge in a data repository [27] [28].

An increasing number of scientific applications are storing and disseminating their datasets using the RDF format [29] [30] [31]. RDF is also being used as an information integration platform in multiple scientific domains. For example, the Biomedical Knowledge Repository (BKR) project at the U.S. National Library of Medicine seeks to create a comprehensive repository of integrated biomedical data from a variety of sources such as biomedical literature (textbooks and journal articles), structured data bases (for example the NCBI Entrez system [32]), and terminological knowledge sources (for example, the Unified Medical Language System (UMLS) [33]). BKR represents the integrated information as RDF statements.

The resources on the Web (including the Semantic Web and eScience) have disparate levels of quality and trust; hence the metadata to describe the origin or lineage of each resource is critical to a range of applications in data mining, knowledge discovery, and data integration. This category of metadata describing the history or lineage of an entity is called *provenance*, derived from the French word *provenir*

that means “to come from.” In addition to their central role in computer science, provenance and trust metadata are also the key enablers for realizing broader objectives in the data driven science as enunciated by the National Institutes Health (NIH) “Translational Research” roadmap that seeks to improve human health by translating “scientific discoveries” (*bench*) to “practical applications” (*bedside*) (Figure 1-1).

We consider three example scenarios to illustrate the importance of provenance information in real world applications:

1. **Patient Health Care**: Provenance information associated with a patient helps doctors and other care givers to arrive at correct decision affecting the treatment and well-being of patients. For example, the original medical condition that led to the patient being admitted, the diagnosis made by specialist doctors, medical procedures performed on the patient, and the response of the patient to treatment plans are all essential provenance information.
2. **Sensor Networks**: In a typical distributed sensor network, sensor observations are collected by sensor of differing capabilities, which are located at different geospatial locations, and may be active during certain time intervals only. These are all examples of necessary provenance information that enable both data processing applications and users to accurately interpret sensor observations in the correct context.
3. **eGovernance**: Many government department release certain types of datasets, for example land survey, water usage, etc., to other agencies for specific and pre-approved applications such as construction of new apartments. To ensure that the released datasets are only used for the intended purposes, the usage history or provenance of the data use is collected and used by the government department. This use of provenance information for ensuring compliance or conformance is essential for regulatory and legal application domains.

1.3 Requirements for Provenance Management in Data Driven Science and the Semantic Web

Provenance metadata has long been used in the cultural heritage domain to trace the origin of a painting, a sculpture or other cultural artifacts [34]. In science, provenance of experiment protocols is often recorded manually in laboratory notebooks and used to verify the quality of data, validate the experiment process, and associate trust value with scientific results. Provenance metadata overlaps with both content dependent and content independent metadata categories (using the classification scheme described in the previous section [15]). In this document, we use the definition of provenance, that is, metadata describing the *history* of an entity to use the temporal value to differentiate provenance from other forms of metadata. In other words, metadata describing past information is categorized as provenance metadata and records the *how*, *where*, *what*, *when, why*, *which*, and *by whom* of an entity.

Provenance has been studied from multiple perspectives in computer science, such as database provenance [35], [36] [37], and scientific workflow provenance [38] [39], but research challenges in provenance management are yet to be addressed. In the following sections, we identify the four major challenges faced in provenance management.

1.3.1 Provenance Representation

A common representation model for provenance will facilitate provenance interoperability, provenance integration across different projects, and enforce consistent use of provenance terminology. A common provenance model should also closely reflect domain-specific details (*domain semantics*) that are essential to answer end user queries in real world applications. Ontologies are considered a suitable modeling solution for these requirements and in addition they also support reasoning to discover implicit knowledge over large datasets [40] [26]. Ontologies are used in many scientific domains [41] and are gaining rapid community acceptance, for example the National Center for Biomedical Ontologies (NCBO) [11] lists 201 ontologies in the life science domain.

Using ontologies to model provenance information will not only reduce terminological heterogeneity to facilitate interoperability, but also support discovery of implicit knowledge over large datasets using reasoning tools.

1.3.2 Provenance Query and Analysis

A variety of provenance queries addressing requirements of a specific application have been used in the literature [42] [43]. To create a flexible and effective query infrastructure for provenance information, we need to understand the specific characteristics of provenance queries. A classification or taxonomy of the provenance queries will help us define focused query operators. A set of provenance query operators will create a standardized query interface with well defined functionalities that can be uniformly implemented across application domains. The provenance query operators will enable users to query provenance information without having to learn different query languages as well as manually compose a complex query pattern. In addition, optimization techniques defined for the standard execution semantics of the query operators can be utilized for provenance queries in different application domains.

1.3.3 Scalable Provenance Systems

The provenance queries in Semantic Web applications, which represent information using the RDF model, are graph traversal operations for path computations. Path computation over graphs is an expensive operation especially in the case of provenance queries that require computation of fixed paths, recursive pattern-based paths and neighborhood retrieval. Further, the large scale of scientific data

increases the query execution costs and adversely affects response time of provenance systems. To enable real world scientific applications to incorporate provenance tracking, practical provenance systems are needed to handle complex provenance queries over very large datasets.

1.3.4 Provenance Tracking on the Semantic Web

The RDF reification vocabulary [24] has been traditionally used by Semantic Web applications to track provenance in RDF documents . The RDF reification vocabulary consists of the four entities `rdf:Statement`, `rdf:subject`, `rdf:predicate`, and `rdf:object`. A variety of problems have been identified in the use of RDF reification vocabulary for provenance tracking in Semantic Web applications including lack for formal semantics associated with the RDF reification vocabulary, a disproportionate increase in total of RDF triples without a corresponding increase in information content, and use of blank nodes [44].

1.4 Dissertation Goals

Existing work on provenance management have often addressed parts of the above listed challenges. The pace of data generation and consumption in science has continued to accelerate and is expected to do so for the foreseeable future. This coupled with the Web based eScience paradigm is a unique opportunity for the research community to conduct transformational science and provenance is the key enabler to realize this vision.

The overarching goal of this dissertation is to define a comprehensive framework for provenance management that defines a common representation model for provenance, a dedicated provenance query infrastructure, a scalable provenance system, and an effective mechanism to track provenance on the Semantic Web. In essence, this framework will enable the creation of practical provenance systems to support scientific discovery in data driven science. We describe the four sub-goals of this dissertation proposal:

- *A common model of provenance* with well-defined formal semantics that facilitates provenance interoperability and is based on Semantic Web standards,
- *A query infrastructure for provenance information* consisting of a classification scheme of provenance queries and a set of query operators to support a variety of provenance queries,
- *A practical scalable provenance system* that scales with provenance queries with high expression complexity and very large scientific datasets, and

- *An efficient and effective provenance tracking mechanism for Semantic Web* applications that does not require the use of RDF reification vocabulary and blank nodes

The research presented in this thesis and related work has been presented in four conference papers [45] [44] [46] [47]; three journal papers [5] [48] [49]; five workshops papers [50] [51] [52] [53] [54]; and two book chapters [55] [56].

1.5 Summary and Dissertation Outline

The dissertation proposal consists of five parts and is organized into the following chapters: **Chapter 2** describes the existing work in provenance management and introduces the notion of Semantic provenance; **Chapter 3** defines the foundational model of provenance to serve as a common provenance representation model; **Chapter 4** describes a provenance query classification scheme and a set of provenance query operators; **Chapter 5** describes the implementation of a scalable provenance query engine; **Chapter 6** introduces a new approach for provenance tracking on the Semantic Web; **Chapter 7** describes the use of the provenance management framework in a real world application; and we conclude and discuss future work in **Chapter 8**.

Chapter 2

Provenance Management: Approaches and Challenges

Introduction

In the previous chapter we discussed the exponential increase in the scale and complexity of experiments made possible by the cyberinfrastructure paradigm. Figure 2-1illustrates a high-throughput scientific workflow, used in the NCRR-funded integrated technology resource for biomedical glycomics project [57], for the processing and analysis of proteomics data using mass spectrometry. The workflow generates hundreds of files per sample run.

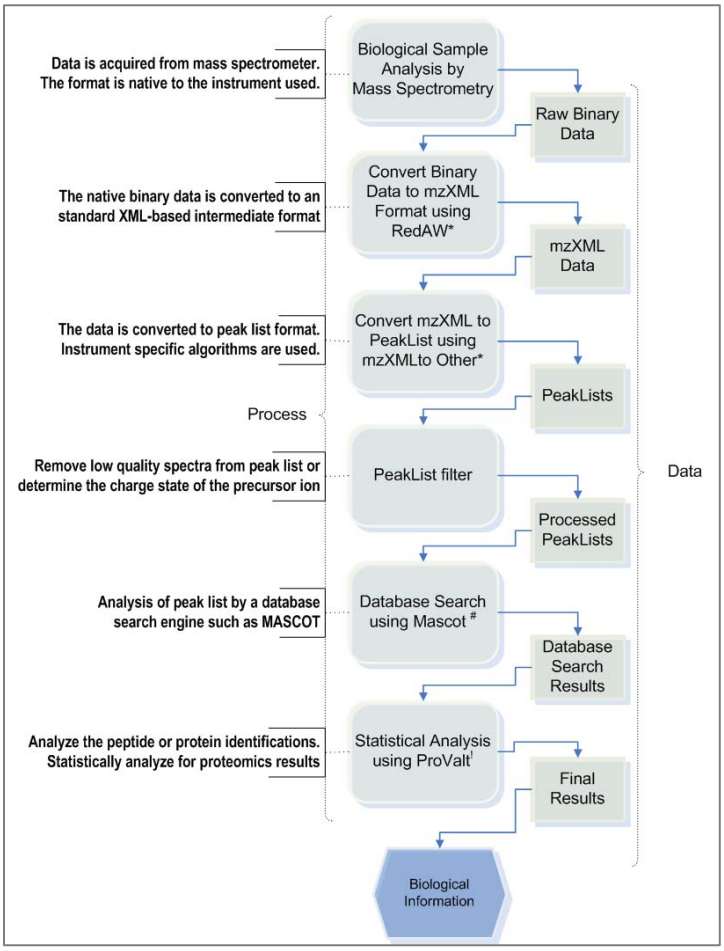


Figure 2-1 Protocol for proteomics data analysis using mass spectrometer

The rapidly increasing volume of data raises multiple issues, such as:

a) How to leverage the data for critical insights that will in turn drive future research.

b) How to seamlessly manage (compare, integrate, and process) large volumes of data generated by hundreds of distributed laboratories using heterogeneous starting materials, equipments, protocols, and parameters

We discussed the role of provenance information in trying to address the above issues in the previous chapter. In the eScience informatics community, sustained research in provenance has led to many models for provenance creation, representation, storage, and querying [38]. Most current eScience approaches to provenance creation center on a "workflow engine perspective of the world," that is, an experiment as viewed by a workflow engine. Hence the *operations* (in the form of Web services or scripts) orchestrated by the workflow engine are the principle actors in the resulting provenance descriptions along with information about the *input files* and *output files*. This approach not only ignores the multiple domain specific relationships that link together data, processes and equipments, but also imposes a system level view on what is essentially a scientific procedure. We term this category of provenance information *system provenance*, also sometimes called as workflow provenance [58].

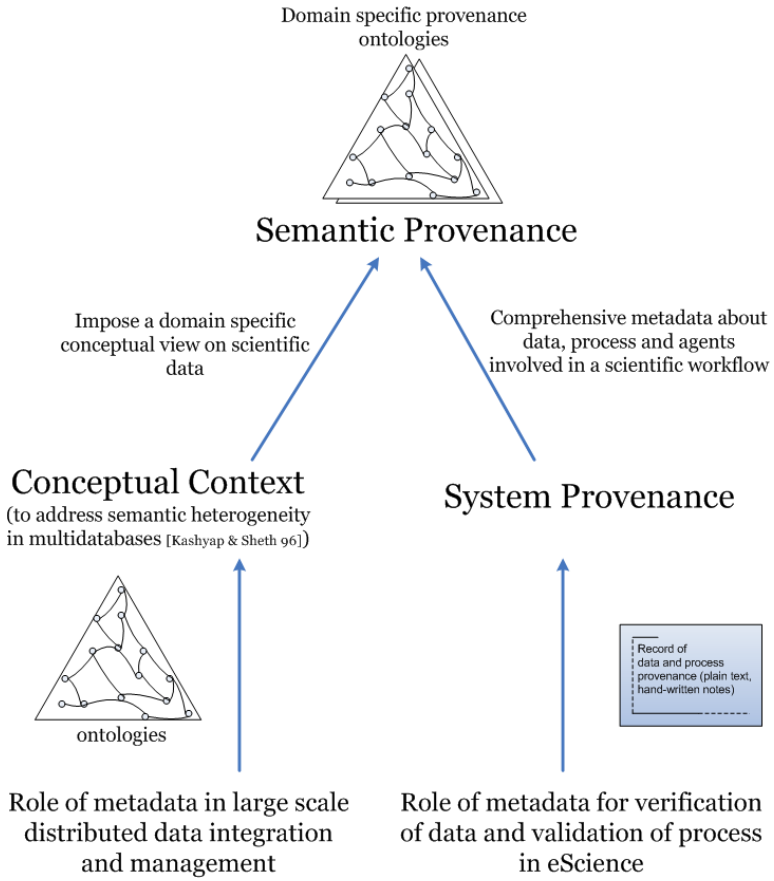


Figure 2-2 Evolution of semantic provenance from role of metadata in (a) data integration in distributed environments; (b) verification of data and validation of processes in eScience

2.2 Semantic Provenance

To be used effectively for management of the growing volume of scientific data, provenance information must be:

a) Expressive: A provenance model should be expressive enough to incorporate explicit domain semantics that closely reflects the complexities of scientific domains. This will facilitate the evolution of a scientific community backed provenance vocabulary and also enables software agents to accurately interpret the provenance metadata.

b) Software-interpretable: Human mediation is inadequate to process, analyze, integrate, store, and query the petabytes of data and associated metadata generated by the industrial-scale processes in eScience. To enable the provenance metadata to be used in eScience data management, it must be "computable" by software agents [59].

To achieve these two objectives, we extend the notion of provenance information and combine it with two important attributes of semantic metadata namely domain knowledge and formal ontological underpinning (Figure 2-2) to define "*semantic provenance*" as follows: "`semantic provenance information is created with reference to a formal knowledge model or an ontology that imposes a domain-specific conceptual view on scientific data. It consists of formally defined concepts linked together using named relationships with a set of rules to represent domain constraints.`"

We illustrate the distinction between system provenance and semantic provenance using a set of queries:

a) *Queries answered using system provenance*: An example is, "Find the original data from which result data X was derived." This query utilizes the workflow-centric provenance information that documents the invocation order of processes, the input data, and the output data for each of the processes. Thus, using the links connecting the output data for a process to its input data, the original data entity for result data X can be traced and identified. Queries in this category are typically used to investigate the protocol that generated the data and to re-run a scientific workflow if needed for validation.

b) *Queries answered using semantic provenance*: Queries in this category are complex and involve relationships that tie data, processes, and instruments together using a domain-specific conceptual view. An example from the proteomics domain is, "Find proteins composed of peptides with N-glycosylation consensus sequence {*N[^P][S/T]*}[1] identified in samples labeled with O18." This query utilizes relationships between data entities that are not modeled in a workflow view of

[1] The regular expression represents an amino acid sequence pattern with Asparagine (N), followed by any amino acid except Proline (P) and then followed by either Serine (S) or Threonine (T). This pattern can be preceded or followed by any number of amino acids.

provenance information such as "a peptide is derived from a protein" and "proteins are identified from a particular sample." Further, the query constrains the samples (introduced in detection equipment such as a mass spectrometer) to be labeled with O18 (an isotope of oxygen), which is again a domain-specific relationship.

Note that in addition to incorporating domain-specific details, semantic provenance can also answer queries of type (a) described above. Hence, semantic provenance features richer structural and semantic details as compared to system provenance. We describe the semantic provenance framework for eScience (Figure 2-3) along three fundamental dimensions:

a) **Semantic provenance creation:** This involves a set of specialized tools plugged into a scientific workflow on demand to create semantic provenance metadata. Extraction of comprehensive metadata from multiple sources, such as generated scientific data and Web forms (e.g., for parameter specifications, equipment details, project details) is another important element of this dimension.

b) **Knowledge models or ontologies:** Domain-specific provenance ontologies to model scientific processes, data (including temporal information), and agents as formally defined concepts linked together using named relationships.

c) **Query, analysis, and visualization:** Using reasoning tools, software agents can process the semantic provenance information to answer complex domain queries. Semantic provenance information will also be used for comparison, integration, retrieval, and visualization of scientific data

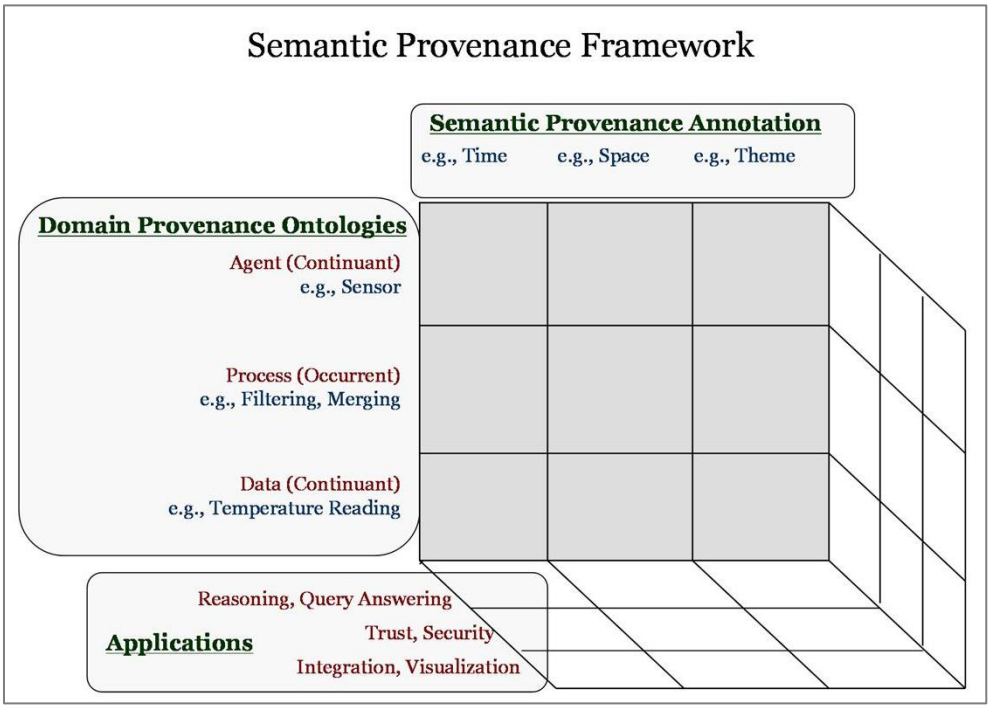


Figure 2-3 The three dimensions of semantic provenance

This Semantic provenance framework supports the important requirements identified by the proposed Open Provenance Model (OPM) [60] (as part of the international provenance challenge) and also addresses many non-functional requirements using the rich set of publicly available resources created by the Semantic Web research community.

Requirements addressed by semantic provenance are:

a) **Provenance information interoperability**: Using ontology schema mapping and merging techniques [61] semantic provenance from different workflows can be shared, integrated and are interoperable.
b) **Ease of application development**: The wide availability of tools for Semantic Web resources such as the Jena toolkit [62] and Sesame [63], make it easier to develop applications
c) **Precise description of provenance**: The well-defined semantics of the RDF model [27], and expressive OWL language enable precise description of provenance information.
d) **Inference capability and digital representation of provenance**: reasoning tools such as Racer [64], Pellet [65], and FaCT++ [66] can be used by software applications to perform reasoning over semantic provenance. Since digital representation is a foundational characteristic of the Semantic Web, semantic provenance supports digital representation of provenance information

Additional advantages offered by semantic provenance:

a) **Publicly available ontologies:** The set of publicly available ontologies listed on open biomedical ontologies (OBO) at NCBO [11] represent a tremendous research effort and should be reused for modeling provenance in life sciences. Many other eScience domains are also developing high quality ontologies such as in geospatial sciences [67] and environmental sciences [68].
b) **Storage and querying resources:** The SPARQL query language [69] has been accepted as W3C recommendation for querying of RDF resources. There are multiple storage solutions available for Semantic Web resources including Oracle 11g [70], Kowari [71], and Virtuoso RDF [72].
c) **Visualization tools for Semantic Web resources:** Many open source applications have been developed for visualization and browsing of Semantic Web data. Some examples projects include Welkin [73], multiple plug-in tools for the Protégé environment [74] and Semantic Analytics Visualization (SAV) [75]

2.3 Conclusions

In this chapter, we introduced the notion of semantic provenance, which unlike database and workflow provenance is expressive enough to incorporate domain semantics and is underpinned by formal ontology modeling to be computable by software agents. We demonstrated that semantic provenance is essential to create a metadata-based framework to manage scientific data, including support for complex user-defined provenance queries in scientific domains. The use of Semantic Web technologies to implement the

semantic provenance allows applications to leverage a large number of existing resources that range from modeling languages such as OWL to storage solutions such as the Virtuoso RDF store.

Chapter 3

Provenance Representation and Provenir Ontology

Introduction

An expressive and extensible model of provenance, with well-defined formal semantics, is essential to support the increasing need for provenance interoperability and integration across research projects and institutions. In addition, a provenance model also needs to incorporate domain-specific details, also known as *domain semantics*, to support user and application requirements in different scientific domains. For example, the provenance queries composed by researchers in biomedicine use terms that reflect the complexities and details specific to the biomedical domain [5]. Hence provenance models, such as workflow or database provenance models, which do not incorporate domain semantics in their provenance representation, are inadequate to create provenance management infrastructure in eScience and related application domains [5]. Further, traditionally provenance in scientific experiments has been represented as hand-written notes to keep track of experiment conditions, results, and project details. In the current scenario of high-throughput experiment protocols that generate extremely large volumes of data, provenance models need to be amenable to automated processing and consistent interpretation by software agents. Ontologies are a suitable modeling solution for addressing these requirements of provenance representation and reasoning to discover implicit knowledge over large datasets [76] [40].

Ontologies are shared vocabularies modeled using a formal language, such as OWL [40], that represent knowledge of a particular domain. Ontologies are used in many scientific domains [11] and as part of the Semantic Web initiative are gaining rapid community acceptance. For example, the National Center for Biomedical Ontologies (NCBO) [11] lists 201 ontologies in the life science domain. Provenance ontologies not only reduce terminological heterogeneity to facilitate interoperability, but also support knowledge discovery tasks through reasoning. ProPreO was one of the first provenance ontologies developed to model provenance in high-throughput proteomics experiments [45].

3.2 ProPreO Ontology: Modeling Provenance in Proteomics Experiments

We developed the ProPreO ontology as a formal representation of proteomics processes and attendant data. The two critical aspects of any '-omics' experiment are the identification of biological entities (*what is it?*) and their quantification (*How much of it is there?*). It is extremely difficult, especially in a high-throughput experiment, to answer these questions by querying datasets that are heterogeneous, developed by multiple researchers who use different methodologies, parameters, and data formats. Although

provenance can form a foundation to compare different datasets and enable researchers to repeat experiments and track the attendant data [77] [78], workflow provenance is inadequate to support such queries [5].

In case of experimental data, the context for comparison and correlation is provided by multiple factors such as the origin of the sample (*e.g.*, malignant or benign tumor cells), experimental methods used in the generation of the data (*e.g.*, the chromatography method used to separate peptides), the settings of individual instruments (*e.g.*, the laser intensity of an ion source), or the database used in identification of peptides. The starting point in the development of ProPreO was the Pedro UML schema [79], which models four stages of experimental proteomics, namely, *Sample Generation*, *Sample Processing, Mass Spectrometry, and MS Results Analysis*. However, the goals of ProPreO are distinct from those of Pedro UML schema [79], and hence these four stages are not defined as top-level concepts in ProPreO. We iteratively evolved the current top level concepts of ProPreO through multiple use cases of applications listed above.

Now, we discuss the top level concepts of ProPreO and illustrate its ability to describe experimental hardware, data processing applications, laboratory tasks, computational tasks, parameter sets, and the resulting data (Figure 3-1). We also discuss its extensibility, which allows new classes of the above listed concepts to be included in the ontology. This reflects the state of real experimental protocols in biology, *i.e.*, they must be sufficiently dynamic to keep pace with new technologies and paradigms.

1. `data` **-** We have modeled the various types of datasets generated at different phases of a glycoproteomics experiment. For example, data generated by analytical techniques, such as mass spectrometry exist in multiple forms. Typically, the 'raw data' initially generated by a mass spectrometry instrument is in a proprietary format. Subsequently, it is processed to generate another subclass of `data`, that is, a list of glycopeptides. The metadata required to provide the relevant experimental context for comparison of processed datasets includes parameter values for the tasks that generated them. The rich set of relationships in ProPreO (Figure 3-1) allows data instances to be compared by associating them with instances of other relevant concepts such as `parameter_lists`.
2. `data_processing_tool` - There are many standard and locally developed software applications used to generate or process data at various stages of the experiment. Metadata semantically annotates data instances, associating each with specific software applications and forming an appropriate context for interpretation and processing.

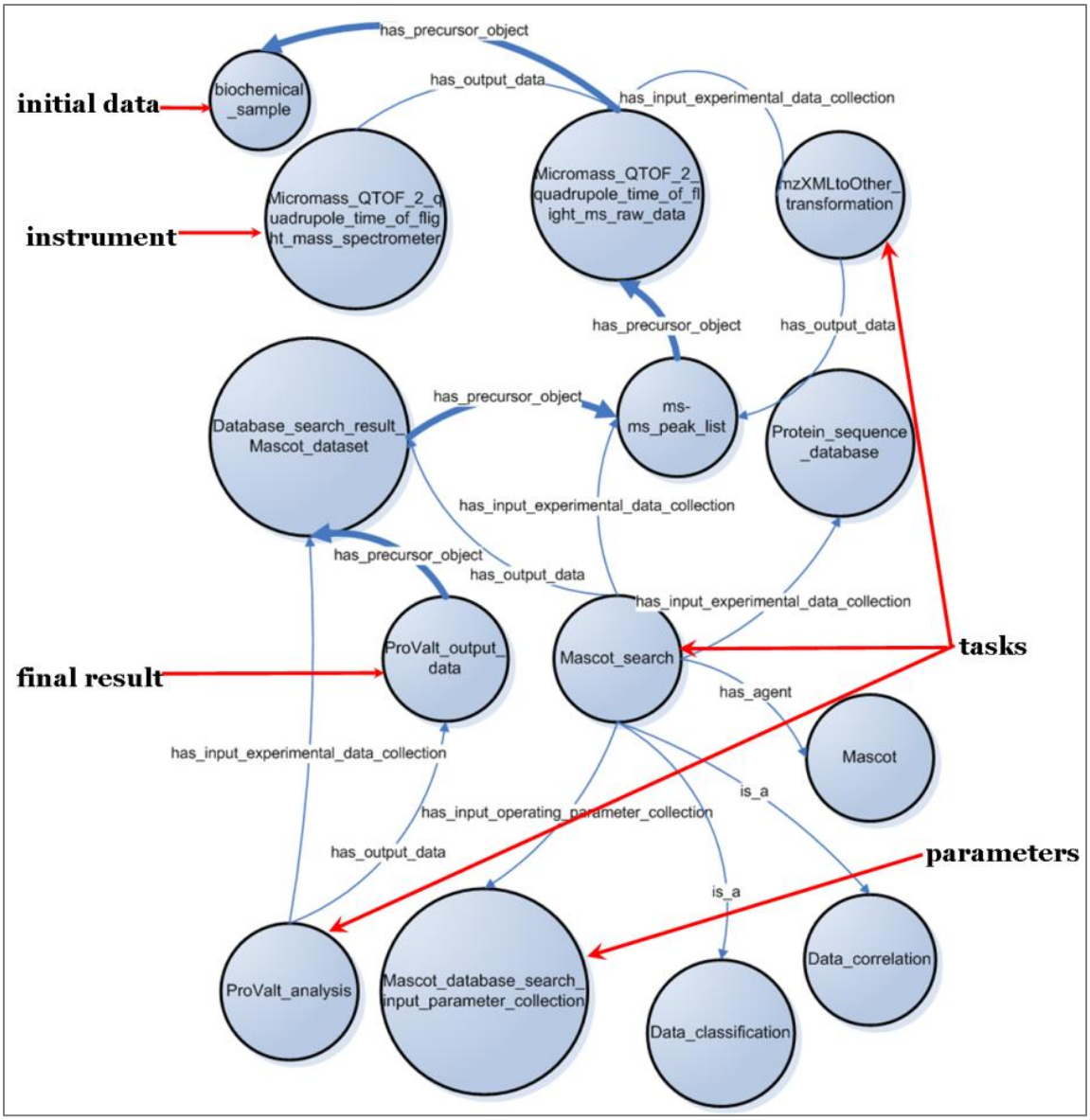


Figure 3-1 A schematic representation of the proteomics data analysis protocol using ProPreO ontology concepts and the named relationships linking them

3. `hardware` - The `hardware` class has two subclasses, namely, `instrument` and `instrument_component`. This enables ProPreO to capture metadata describing the states of the various components of the instrument that generated a given instance of data. This metadata is also necessary to determine whether two datasets can be directly compared. For example, the instrument component `HPLC_diode_array_detector` has two properties that specify the range of wavelengths that are accessible to the detector (`has_wavelength_detection_max` and `has_wavelength_detection_min`). These properties allow a reasoner to infer that data

generated using an ultra-violet (UV) detector cannot be directly compared with data generated using a visible light detector.

4. `molecule` - This defines the very broad classes of molecules that are analyzed in a proteomics experiment; for example, glycans, proteins and peptides. These classes are themselves defined (at least in part) by their equivalence to analogous classes in more specialized ontologies such as GlycO [45]. ProPreO extends these class definitions by adding properties that provide an experimental context of reference. This framework allows, for example, a `peptide` to be associated with its `experimental_chemical_mass`, a property that may not be defined in the referenced ontology.
5. `organism` - This class describes the taxonomic classification of a biological species, again by reference to a more specialized ontology, providing the biological context of a given sample instance.
6. `parameter_list` - Each instance of data has a set of parameter values that are associated with its generation. These include the instrumental settings, environmental parameters and variable setup parameters used by software applications to process data. ProPreO models parameters relating to database searching, HPLC runs and mass spectral analysis. The experimental context of a `parameter` (which is a type of `data`) can be inferred by its inclusion in a particular `parameter_list`. Parameters are further classified according to their relationships to specific experimental `task`, providing a rich framework for analyzing their relevance with regard to experimentally obtained data.
7. `task` - A glycoproteomics experiment can be viewed as a set of human-mediated or automated tasks that generate real-world samples or data to be used as input for the next step.

ProPreO currently has approximately 490 classes and 35 named relationships with 145 constraints, such as class-level restrictions. It is also populated with 3.1 million instances of `propreo:tryptic_peptide`. ProPreO has been released for community use and is listed at the OBO at NCBO.

Similar efforts in creating provenance ontologies include the Microarray Gene Expression Data (MGED) ontology that was created to track provenance in microarray experiments [80], while the Functional Genomics Investigation Ontology (FuGO) was created to model provenance in functional genomics. These multiple projects in provenance management highlight the need for interoperability of provenance information, which can be facilitated if the provenance ontologies share a common modeling basis and uniform use of terms. Foundational *upper-level ontologies* have traditionally been used to enforce these consistent design principles and to facilitate domain ontology interoperability and integration [81].

In the next section, we describe the creation of the provenance upper-level ontology called Provenir.

3.3 Provenir Ontology: An Upper-level Provenance Ontology

The primary challenge in defining the structure of the Provenir ontology is to strike a balance between the abstract upper-level ontologies, such as the Suggested Upper Merged Ontology (SUMO) [82], and the detailed domain-specific provenance ontologies. The Provenir ontology represents the set of provenance terms that are common across domains and can easily be extended by developers of domain-specific provenance ontologies according to their requirements. This not only significantly reduces the workload of ontology developers, but also ensures modeling consistency.

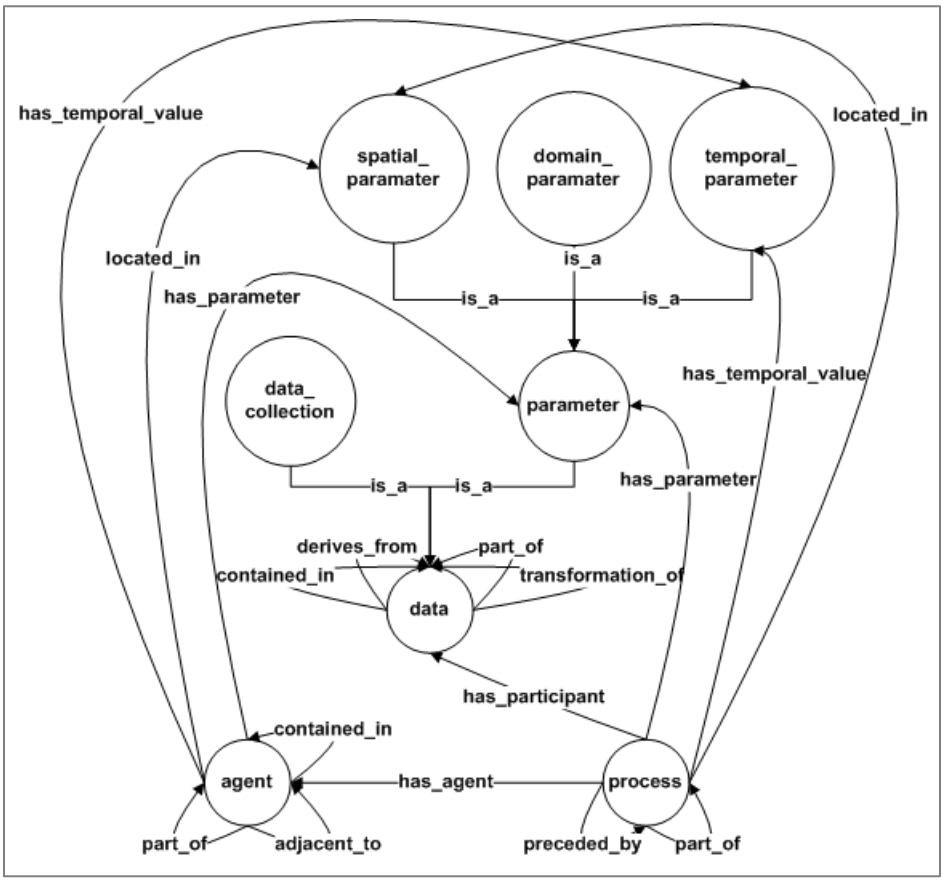


Figure 3-2 Provenir ontology schema

The top-level classes in the Provenir ontology are derived from two primitive concepts of "occurrent" and "continuant" defined in philosophical ontology [83]. Continuant represents "… *entities which endure, or continue to exist, through time while undergoing different sorts of changes, including changes of place*" [83] and occurrent represents *"...entities that unfold themselves in successive temporal phases*" [83]. It is easy to see that occurrent corresponds to the "process" entities used in science, such as experiment processes and data processing. Other entities can be categorized into two sub-types of continuants, namely, "agents" such as temperature sensor and "data" such as visualization charts. The

three concepts of process, data, and agent form the top-level classes in the Provenir ontology (Figure 3-2) representing the primary components of provenance. The two base classes, "data" and "agents" are defined as specializations (sub-class) of "continuant". We present the definition of each class as follows:

1. data: This class models continuant entities that represent the starting material, intermediate material, end products of a scientific experiment, and parameters that affect the execution of a scientific process. The class data inherits the properties of continuants such as enduring or existing while undergoing changes.
2. process: This class models the occurrent entities that affect individuals of data (for example, process, modify, create, and delete among other dynamic activities).
3. agent: This class models the continuant entities that causally affect the individuals of process.

In addition to these three base classes, the Provenir ontology also differentiates between the data values that undergo change in a process, for example, conversion of sensor data to create visualization charts, and parameter values that influence the behavior of a process (Figure 3-2):

a) data_collection: This class represents atomic or composite data entities that are acted upon during a process.
b) parameter: parameter is a class of values that affect the behavior of a process or agent as constraints.

There are three subclasses of parameter, defined along the three dimensions of space, time, and theme (domain-specific):

i) temporal_parameter: This class captures the temporal details associated with individuals of data_collection class (for example, the timestamp associated with a sensor reading), process (for example, the duration of a protein analysis process), and agent (for example, the time period during which a sensor was working correctly).
ii) spatial_parameter: The spatial_parameter associated with individuals of process or agent or data_collection classes is represented by this class. The geographical location of an ocean buoy is an example of spatial_parameter.
iii) domain_parameter: The domain_parameter class is used to model domain-specific parameters (for example, tolerable salinity levels for ocean buoys).

Further, the explicit modeling of relations as first class entities is an important characteristic of Semantic Web modeling languages. For the Provenir ontology we adapted the properties defined in the Relation ontology (RO) [83] to link provenance terms. The RO was created by the Open Biomedical Ontologies (OBO) Foundry. RO defines a set of ten primitive properties with well-defined "domain" and

"range" values. These properties were mapped to appropriate Provenir classes with restrictions on their domain and range values as required for provenance modeling. For example, the class `data` is linked to a process as either input or output value, and this is modeled by the relation "`has_participant`" (*domain*: `process`, *range*: `data`). Similarly, the notion that an `agent` initiates, modifies, or terminates a `process` is captured by the relation "`has_agent`" (*domain*: `process`, *range*: `agent`). We note that the class `data` is linked to `agent` only through an intermediary `process`, for example, a list of peptides (`data`) is linked to a mass spectrometer (`agent`) through the protein analysis `process`. In the next sections, we describe the properties of the Provenir ontology (Figure 3-2).

1. `part_of` – This property is defined for each of the three base classes of Provenir ontology. The restriction for this property is that the domain and range values belong to the same class. For example, if `data` is the domain (range) respectively of the properties, the corresponding range (domain) respectively is also `data`. As defined in the RO [83], this property satisfies the standard axioms of mereology, that is, reflexivity, anti-symmetry, and transitivity.
2. `contained_in` – The `contained_in` property is also defined with similar constraints as `part_of`, that is, the domain and range values belong to same class and do not overlap. The property is defined for `data` and `agent` classes. Consistent with its definition in RO, the property is also defined to be non-transitive.
3. `adjacent_to` – This property is defined for disjoint continuants in RO. In Provenir ontology, it is defined only for agent class, where the adjacent spatial location of individuals of `agent` class may have an effect on data values. For example, presence of a sensor generating a magnetic field may affect the quality of observations made by an adjacent sensor. We note that, similar to the mereotopological relations defined in RO [83] such as partial overlap, tangential proper part etc., corresponding properties can be added to extensions of Provenir ontology.
4. `transformation_of` – RO defines `transformation_of` as a property between two entities that preserve their identity between the two transformation stages. For example, let there be an individual that belongs to a subclass of data, say d_1, at time t_1; and the individual also belongs to another subclass of data, say d_2, at a time t_2 with $t_1 > t_2$. If at no time instant, the individual is a member of both d_1 and d_2, then there is a `transformation_of` property linking the two individuals.
5. `derives_from` – This RO property represents the derivation history of data entities as a chain or pathway. Unlike `transformation_of` property which links identical entities, `derives_from` links distinct individuals of `data`. For example, a peptide sample is derived from a protein sample.

6. `preceded_by` – This RO temporal property is defined for distinct individuals of `process` class. Similar to its interpretation in [83], more specific types of properties, such as "*immediately_preceded_by*" [83] with more precise semantics may be defined in extensions to Provenir ontology.
7. `has_participant` – This is the primary property linking `data` to `process`, where the individual of `data` class participates in a `process`.
8. `has_agent` – This is a causal property that links `agent` to `process` where the `agent` is directly responsible for the change in state of the `process`. Similar to the description used in [83], the Provenir ontology also allows the use of this property to "capture the directionality" of scientific experiments, for example, to capture which `agent` caused the activation of a `process`.
9. `has_parameter` – This property is not present in RO [83]. It links the individual of class `parameter` to an individual of the classes `data_collection`, `agent`, and `process`.

Two sub-properties of `has_parameter` describing the temporal and spatial parameters are also defined:

i) `has_temporal_value` – This is a specific property to assign temporal value to individuals of `data_collection`, `process`, and `agent` classes. For example, d_1 `has_temporal_value` t_1.

ii) `located_in` – An instance of `data` or `agent` is associated with exactly one spatial region that is its exact location at a given instance of time. In Provenir ontology, this relation has two domain classes `agent` and `data_collection` with `spatial_parameter` as the range class.

The Provenir ontology is represented in OWL DL, a decidable profile of the OWL [26], with a description logic expressivity of $\mathcal{ALCH}$. This ontology is under active consideration at the OBO Foundry for listing as a foundational model of provenance.

3.4 A Modular Approach for Creating Provenance Ontologies

Domain-specific details are an important component of provenance metadata. But, a single monolithic provenance ontology that models all possible details from different domains is clearly not feasible. Hence, our proposed modular approach involves integrated use of multiple ontologies, each modeling provenance metadata specific to a particular domain (for example, the Open Biomedical Investigation (OBI) ontology represents biomedical domain-specific provenance [80]). These multiple ontologies will use the Provenir ontology as the common reference model, hence making it easier to interoperate with each other. This modular framework represents a scalable and flexible approach to provenance modeling that can be adapted to the specific requirement of different domains. The Provenir ontology forms the core

component of a modular approach to our provenance management framework. We also demonstrate that the Provenir ontology is a well-engineered foundational ontology that can be easily extended.

3.4.1 Parasite Experiment Ontology: Provenance Tracking in Parasite Research

The Parasite Experiment (PE) ontology was developed as part of the NIH-funded *T.cruzi* Semantic Problem Solving Environment (SPSE) project [46]. The PE ontology extends the classes and relationships in *provenir* ontology to model provenance information associated with "Gene Knockout" (GKO) and "Strain Creation" (SC) experiment protocols (Figure 3-3). The GKO and SC protocols consist of multiple sub-processes, which are modeled in PE ontology as `sequence_extraction`, `plasmid_construction`, `transfection`, `drug_selection`, and `cell_cloning` classes (Figure 3-3).

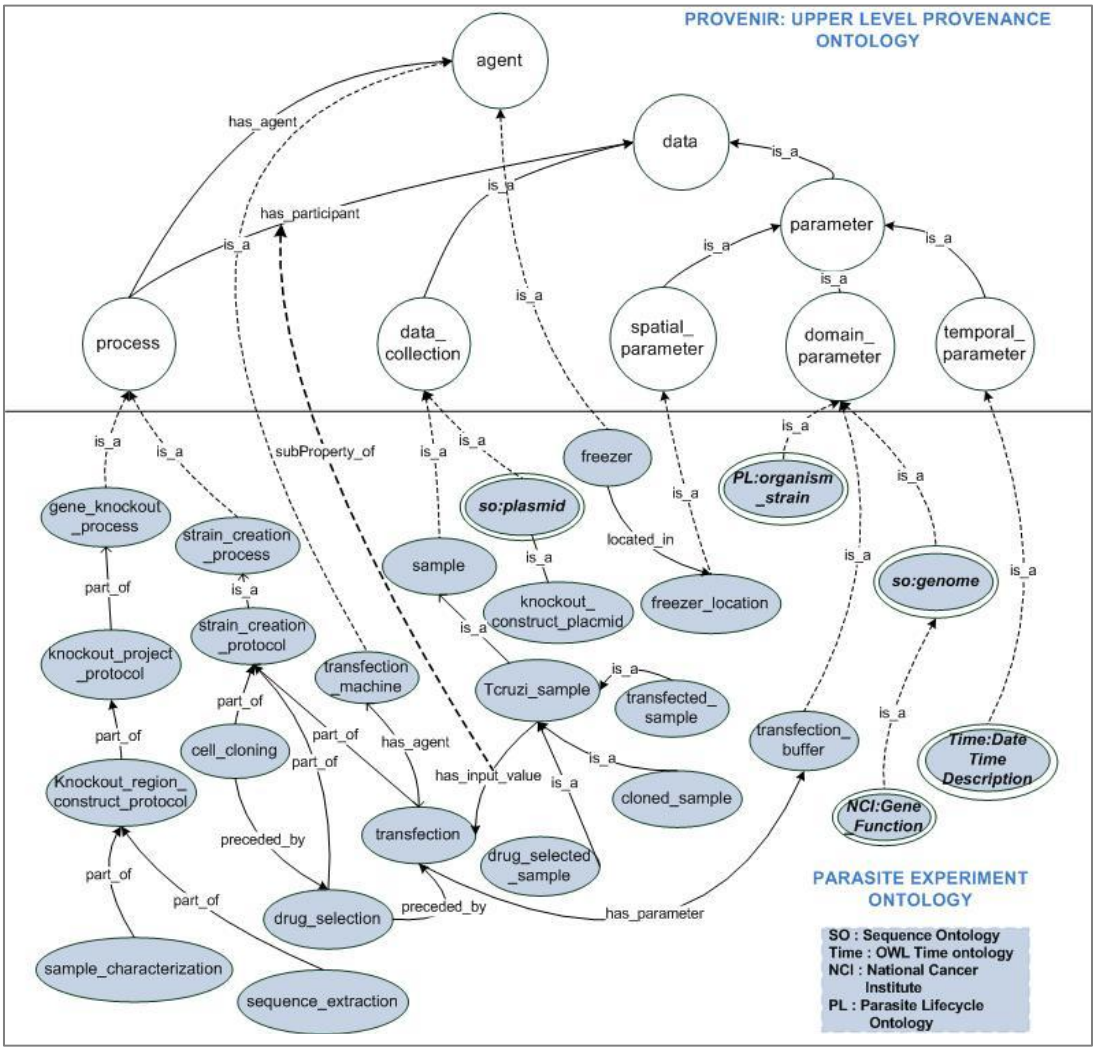


Figure 3-3 Extending Provenir ontology to create Parasite Experiment ontology

First, we discuss the modeling of process entities that constitute the GKO and SC experiment protocols. Two classes namely, `gene_knockout_process` and `strain_creation_process`, are created as subclass of `provenir:process` class, to model generic gene knockout and strain creation experiment processes. The `knockout_project_protocol` and `strain_creation_protocol` classes represent the particular protocols used in the Tarleton research group [46]. The GKO and SC protocols consist of multiple sub-processes, which are also modeled in PE ontology, for example, `sequence_extraction`, `plasmid_construction`, `transfection`, `drug_selection`, and `cell_cloning` (Figure 3-3).

Next, we describe the PE ontology concepts that model the datasets and parameters used in the GKO and SC experiment protocols. Similar to the Provenir ontology, the PE ontology models data and parameter values as distinct entities, for example, given the transfection process its input value `Tcruzi_sample` is modeled as a subclass of `provenir:data_collection` whereas the `parameter` value `transfection_buffer` is modeled as a sub-class of the `provenir:parameter`. Further, the `parameter` values in PE ontology are categorized along the space, time, and theme (domain-specific) dimensions (Figure 3-3).

The third set of PE ontology concepts extend the `provenir:agent` class to model researchers and instruments involved in an experiment. For example, `transfection_machine`, and `microarray_plate_reader` are instruments modeled as subclasses of `provenir:agent`; `researcher` is an example of human `agent`; and `knockout_plasmid` is an example of a biological `agent`. Finally, we describe the properties used to connect the PE ontology classes. In addition to the eleven relationships in Provenir ontology, new object and datatype properties specific to GKO and SP experiment protocols were created. For example, four new object properties are defined to model the similarity relationships between two genomic regions, namely, `is_paralogous_to`, `is_orthologous_to`, `is_homologous_to`, and `is_identical_to`. The PE ontology is modeled using the OWL-DL language with a DL profile of $\mathcal{ALCHQ(D)}$, and contains 118 classes and 23 named relations. PE ontology is open sourced through the NCBO.

3.4.2 Janus Ontology: Modeling Workflow Provenance in Taverna Scientific Workflow Engine

In collaboration with researchers working on Taverna [84], arguably one of the most popular open-source scientific workflow engine at the University of Manchester, we extended the Provenir ontology to create the Janus ontology for modeling workflow provenance [50]. The existing Taverna provenance model defined in [50] consists of a static component describing the workflow specification (processors,

processor ports, and data dependencies as links between ports) [50], while the dynamic component of the provenance model represents entities associated with the workflow execution [50]. As illustrated in Figure 3-4, the classes `janus:workflow_spec`, `janus:processor_spec`, and `janus:port`, are used to model the static component of the existing Taverna provenance model [50].

Similar to the approach used in the PE ontology, the Janus ontology re-uses the classes defined in four public ontologies listed at NCBO, namely, the BioPAX, National Cancer Institute Thesaurus, Foundational Model of Anatomy (FMA), and the Sequence ontologies, and the fifth ontology, OWL Time available from the W3C. This approach allows the interoperability of provenance graphs generated as part of the Janus framework with existing biological data resources such as KEGG [85], Reactome [4], and BioCyc [86]. As part of our ongoing collaboration, we plan to extend the Janus ontology with domain terms used to annotate the default set of services in the Taverna release version [50].

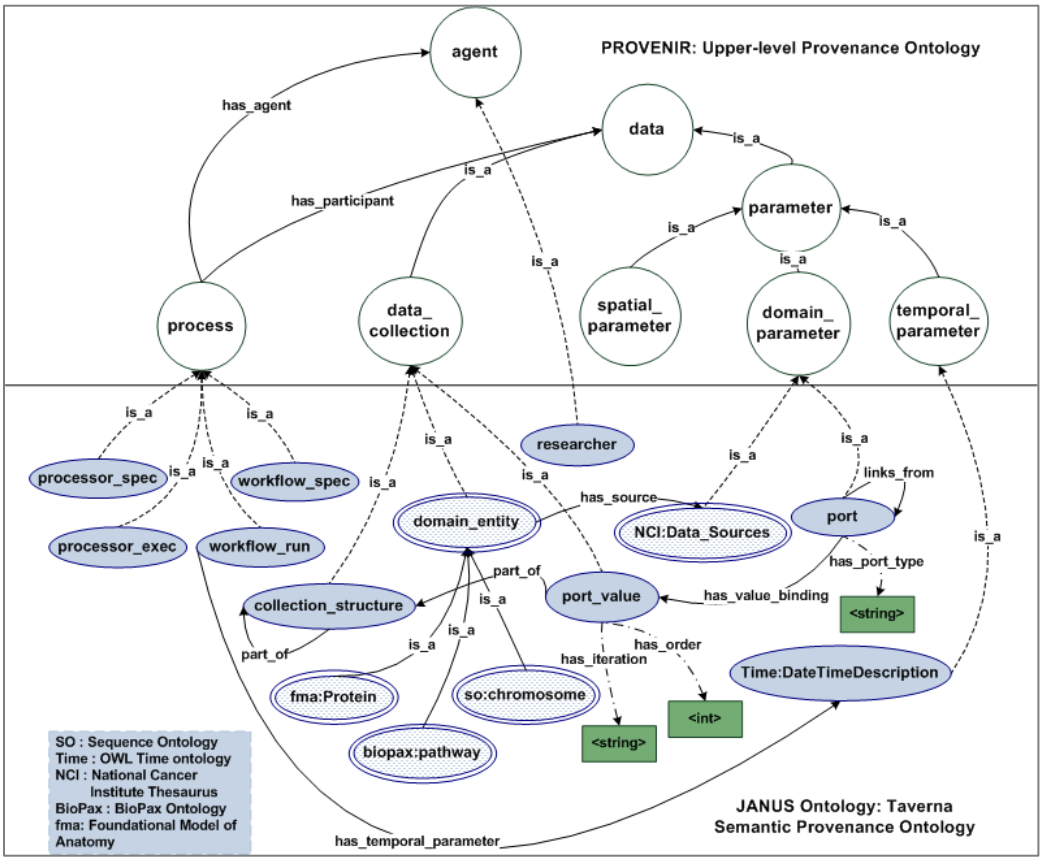


Figure 3-4 Extending Provenir ontology to create Janus ontology for Taverna workflow engine

3.4.3 Trident Ontology: Provenance Tracking in Oceanography

The provenance information for the Neptune project can be divided into two categories of: (a) workflow provenance corresponding to the components of a scientific workflow, and (b) domain-specific

provenance capturing the domain semantics such as details of the sensors and ocean buoy. Existing provenance systems have primarily focused on only workflow provenance [87] while partially or completely ignoring *domain semantic*s. For example, in the Neptune oceanography project (discussed in Chapter 5) the description of the sensors and the location of ocean buoys are critical provenance information.

The Provenir ontology and its extension in the form of the Trident ontology incorporate both the domain semantics and the workflow-specific provenance. The Trident ontology extends the Provenir agent class to model the temperature and ocean current sensors as well as the ocean buoy. Further, the sensors are linked to specific ocean buoys using the `contained_in` relation that enables tracking of sensor data from the damaged ocean buoys. Figure 3-5 gives an overview of the Trident ontology schema and maps its classes to the Provenir ontology.

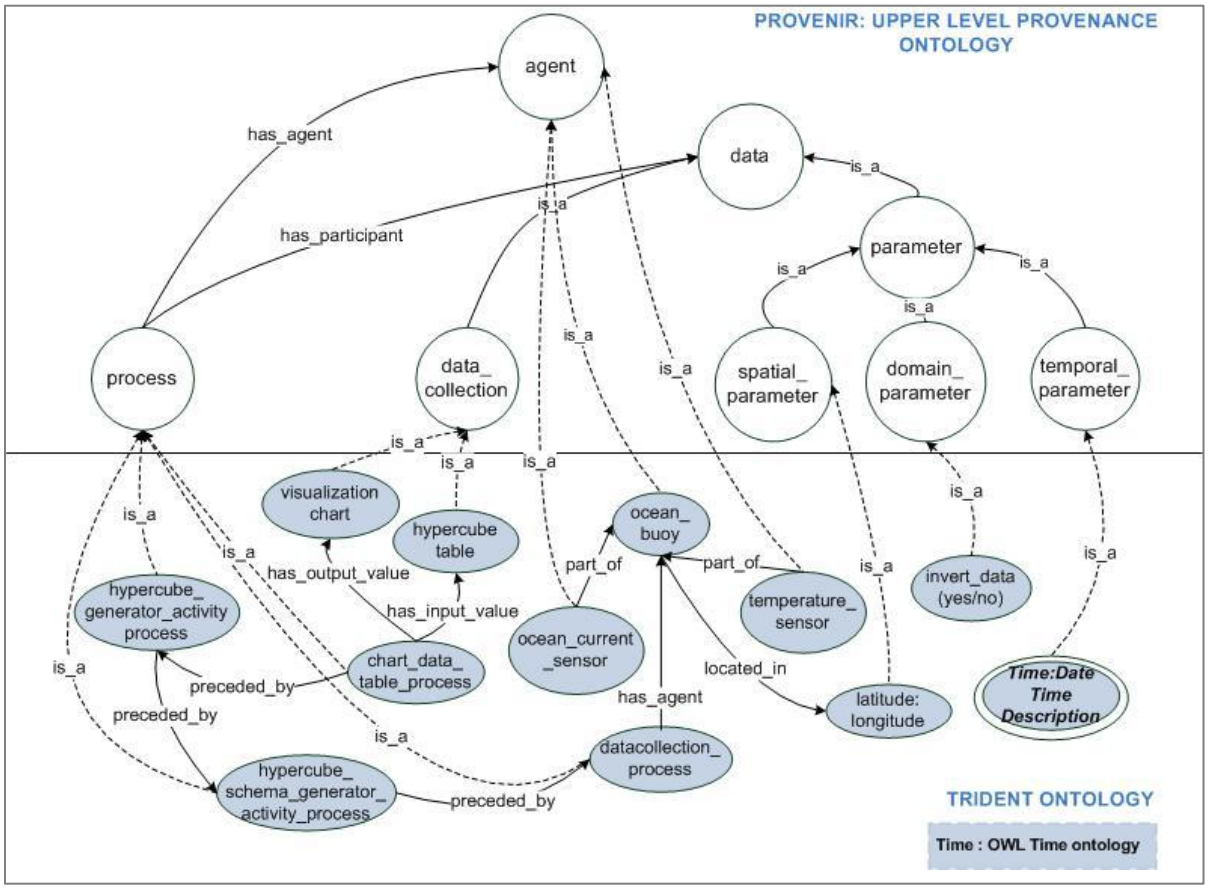


Figure 3-5 Extending Provenir ontology to create Trident ontology for oceanography

The formal OWL-based representation of provenance information in the Trident ontology enables the use of the rich set of Semantic Web reasoning mechanism [76] for provenance query and analysis.

3.5 Related Work

Multiple provenance representation models have been proposed including the Open Provenance Model (OPM) [88], the provenance vocabulary [89], and the proof markup language [90]. Provenance representations, in the context of relational databases, extend the relational data model with annotations [91], provenance and uncertainty [92], and semirings of polynomials [37]. Provenir ontology can be extended to model the provenance of tuple(s) in relational databases, which relies on mappings defined between description logic to relational algebra [93]. In case of OPM, similar to the Provenir ontology it defines "agent", "process" and "artifact" as the three top level concepts. But, in contrast to the Provenir ontology that models 11 fundamental named relationships, OPM (core specification) models only "causal relations" linking provenance entities [88]. This is a significant drawback of OPM that makes it difficult to model partonomy, containment, and non-causal participatory provenance properties needed in many eScience applications (discussed earlier in Section 3.4). The provenance vocabulary, unlike Provenir ontology, has been specifically created to model provenance in Linked Open Data (LOD) applications [89] but does not incorporate domain semantics, which is essential for using provenance information in an application. The Provenir ontology has been extended to create a provenance ontology for sensor observations being published as part of the LOD cloud [51].

In addition to drawbacks related to modeling of provenance information, the OPM is defined as a generic graph structure without an underlying formal representation language [94]. Hence, unlike the Provenir ontology, it does not support a set of standard entailment rules (for example RDFS [27] or OWL inferencing rules [40]. Further, the inference rules defined in the OPM core specification are limited and the OPM specification does not define a clear mechanism for defining additional rules that may be required by an application domain [88]. In contrast, the Provenir ontology takes advantage of the Semantic Web inferencing layer [76] to support the definition of any domain-specific inference rules.

3.6 Conclusions

In this chapter, we address the challenge of defining an expressive model for provenance representation that incorporates both domain-specific details and has well-defined formal semantics. We first described the ProPreO ontology, one of the first domain-specific provenance ontology to model provenance information for proteomics domain. Extension of the ProPreO ontology to model provenance in other application domains was not practical hence we defined a modular approach for provenance representation in various domains underpinned by a common reference ontology. We created the Provenir ontology as an upper-level ontology for provenance representation. The Provenir ontology models a minimal set of common provenance concepts and relationships that can be extended to create domain-specific provenance ontologies in multiple scientific domains. The suite of domain-specific provenance

ontologies use uniform modeling patterns, which is consistent with the Provenir ontology, thereby facilitating provenance interoperability. The extensibility of the Provenir ontology is demonstrated through the creation of three domain-specific provenance ontologies in the *T.cruzi* biomedical informatics project for parasite research, the Taverna workflow system, and the Neptune oceanography project.

Chapter 4

Provenance Query and Analysis

Introduction

In addition to provenance representation, a well-defined and expressive query mechanism is essential to enable real world applications to effectively analyze provenance information. The provenance literature discusses a variety of queries that are often executed using generic or project-specific query mechanisms that are difficult to re-use. Provenance queries in workflow systems focus on execution of computational process and their input/output values (for example, the queries featured as part of the Provenance Challenge [95] are focused on workflow systems). Provenance queries in relational databases trace the history of a tuple or data entity [96]. In contrast, scientists formulate provenance queries that incorporate domain semantics using application-specific terminology [5]. Further, domain scientists are often not familiar with query language such as SQL and face difficulty in composing effective provenance queries. Provenance queries discussed in the literature [37, 97, 98] are characterized by complex query patterns. For example, queries in SQL are recursive as well as nested [99, 100]. In addition to the query pattern complexity, the increasingly large size of scientific datasets incorporating provenance information [101] make provenance query optimization an area of active research [102] [103].

We argue for a dedicated provenance query infrastructure that, (a) supports provenance queries using domain-specific vocabulary, (b) can be re-used in different provenance applications, (c) supports composition of provenance query patterns without requiring scientists to learn complex query languages, and (d) support optimizations for efficient execution of provenance queries over both large datasets and complexity of query patterns. We address the first three issues, that is, (a), (b), and (c), in this chapter and discuss the approaches for optimization of provenance queries in Chapter 5.

Provenance queries have characteristics that can be exploited for creating an effective query infrastructure to support queries of multiple domains and projects. We defined one of the first classification schemes for provenance queries [43] that used the input/output values of the queries to categorize provenance queries. Further, we use the provenance query classification scheme to define a set of provenance query operators. The provenance query operators are defined in terms of the Provenir ontology, which enables the query operators to be used with any domain-specific provenance ontology that extends the Provenir ontology. First, we discuss the characteristics of provenance queries and then introduce the provenance query classification.

4.2 Provenance Query Characteristics and Classification Scheme

We characterize provenance queries based on the input/output values of the queries. For example, a common query is to retrieve the provenance information associated with a given entity (*input* = entity, *output* = provenance trace). If we interchange the input and output values of a query (*input* = provenance trace, *output* = entity), another type of query can be identified that retrieves a set of entities that satisfy a set of provenance information assertions. In addition to these two categories, we identify two other categories of provenance queries for manipulating (comparing and combining) provenance information in our classification scheme:

- ***Category I*: Queries to Retrieve Provenance of an Entity**: Given a data entity, this category of queries retrieves the complete provenance information that influenced the current state of the data entity. This category of provenance queries is characterized by a transitive closure operation. If the provenance information is represented as a graph, a path is computed from the input entity, through intermediate data entities and processes, to the initial or starting point of the provenance graph. This path, representing the provenance of the input value, may either be composed of only data and processes that are directly linked to the input value or may contain additional provenance information such as parameter values, type of instruments etc. used in an experiment.

  An example query in the biomedical informatics domain is the retrieval of the provenance of a new "*stem cell line*". The result will include the starting material, the conditions under which the experiment was conducted, the experiment protocol that was followed, the instruments used, and the researcher(s) who conducted the experiment.
- ***Category II*: Queries to Retrieve Entities (satisfying provenance constraints)**: These queries retrieve entities that satisfy a given set of provenance constraints. The provenance constraints can be interpreted as contextual metadata that is used to identify a subset of entities and are collectively termed as "provenance context". The results of this category of queries are interpreted to have been generated or processed under (a) similar or (b) equivalent conditions depending on the *specificity* of the provenance context structure.

  An example query from the Neptune oceanography project [104] is the retrieval of "*chart visualization*" files created using data from a damaged ocean buoy with identifier "*oceanBuoy7044*" located at the geographical coordinates "*475111N:1222118W*" between "*April 21, 2003 and May 2, 2003*". This class of queries does not figure frequently in the provenance literature and has been recently categorized as being more complex than the first category of queries [100].
- ***Category III*: Queries to Compare the Provenance Traces**: Comparison of provenance traces of two (or more) entities is the third category of provenance queries. The provenance traces can be tested

for identity, inequality, or checked for various degrees of similarity. (Note that graph isomorphism and sub-graph isomorphism problems are NP-complete).

An example query of this category is the comparison of two temperature sensor observations, from the Neptune oceanography project [104], which requires the comparison of the associated provenance information to verify that they were created under equivalent experimental conditions and generated by the same type of sensors.

- ***Category IV*: Merging of Provenance Traces**: In many applications domains, entities are generated in multi-step processes or phases. For example, in parasite research [105], an avirulent strain of a parasite is created using two experiment protocols, namely Gene Knockout and Strain Creation. During the Gene Knockout experiment, gene(s) are "knocked out" to create "knockout construct plasmids", which are subsequently used in the Strain Creation experiment to create a new strain of the parasite. To identify the particular gene(s) that were "knocked out" to create a specific strain of the parasite requires the merging of the provenance traces from the two experiments [46].

The four broad categories of provenance queries can also be further sub-divided into more specific types of queries. For example, Category I queries can be restricted to retrieve provenance information describing instruments used in a biomedical experiment. This sub-division is driven by application requirements and will vary across projects and domains. There is more recent work in characterizing provenance queries. For example, [100] characterizes the Category I (provenance retrieval) and Category II (data retrieval) as "backward" and "forward" provenance queries.

In addition to facilitating a better understanding of provenance query characteristics, we use the classification scheme to create a practical provenance query infrastructure consisting of a set of specialized query operators and a query engine that efficiently implements the query operators.

4.3 Provenance Query Operators

We introduced the provenance query operators as a standard query mechanism that can be consistently implemented in provenance management applications. The query operators also allow users to leverage their well-defined semantics to issue queries without having to repeatedly compose complex query patterns. We first discuss the provenance model for the query operators followed by a description of their syntax and semantics.

4.3.1 Using the Provenir ontology as the Provenance Model

The provenance query operators are defined in terms of the Provenir ontology schema (discussed in the previous chapter) and are executed over instance data represented as a RDF document [24]. The former

scheme corresponds to the description logic Terminological Box (TBox) and the latter corresponds to Assertional Box (ABox) respectively [106]. The provenance RDF graph is a grounded graph with no blank nodes. These graphs have many disadvantages including lack of any global meaning outside the RDF document [27]. Further, the values in the provenance RDF graph are "typed" using the `rdf:type` property in terms of concepts and relations either in the Provenir ontology or a domain-specific provenance ontology (that extends Provenir ontology). In case a domain-specific provenance ontology is used to "type" instance values, the standard RDFS entailment rules [27] allows applications to infer the corresponding Provenir ontology concept or property (by following the `rdfs:subClassOf` or `rdfs:subPropertyOf` properties).

Hence, the definition of the provenance query operators in terms of the Provenir ontology allows the query operators to be executed over any provenance RDF graph that conforms to a domain-specific ontology that extends Provenir ontology. This enables the provenance query operators to be seamlessly ported to different application domains. We use the standard RDF graph syntax and semantics [27] to define the provenance query operators. Further, we use a quadruple consisting of vertices, edges, mapping function to label vertices and edges, and a mapping function to map vertices to Provenir classes, to describe a provenance graph.

$G = (V, E, l, m)$
$E \subseteq V^2$
$l: V \cup E \rightarrow L$ (L is set of Uniform Resource Identifiers)
$m: V \rightarrow Nm$ (Nm is set of Provenir ontology classes)

4.3.2 *provenance ( )* Query Operator

The first query operator is defined to support ***Category I*** provenance queries and is a closure operation on the provenance information. The operator takes as input any given data entity and returns the complete provenance information associated with the data entity. The *provenance ()* query operator defines a pattern constructed in two steps:

(a) **Initialization phase**: The individuals of the class `process` linked to the input value by property `has_participant` (Figure 4-1) are added to the intermediate result set.

(b) **Recursive phase**: The `process` individuals are used to locate all individuals of the class process, linked by the property `preceded_by`, which is implemented as a transitive closure function for the < `process`, `preceded_by`) class-property pair of entities. Further, all individuals of the `agent`, `parameter`, and `data_collection` classes linked to process individuals are added to the query result (Figure 4-1).

The query operator is defined formally using set-theoretic representation, where the notation <*s*, *p*, *o*> stands for a subject, predicate, object triple as used in a provenance RDF graph.

Definition 1: *provenance ()*

SS: Search Space (set of all available triples)
Proc = {p | (p, rdf:type, process) ∈ SS}
Agent = {a | (a, rdf:type, agent) ∈ SS}
DataC = {d | (d, rdf:type, data) ∈ SS}
Input: dc

IN = {t ∈ SS | t = (s, has_participant, dc)}

P is smallest set such that,
P = ({p | ∃ x, y. (p, x, y) ∈ IN} ∪ {p | ∃ x ∈ P. (x, preceded_by, p)}) ∩ Proc
A = {a | (p, has_agent, a) ∈ SS ∧ p ∈ P} ∩ Agent
D = {d | (p, has_participant, d) ∈ SS ∧ p ∈ P} ∩ DataC
OUT = IN ∪ {(p_1, preceded_by, p_2) ∈ SS | p_1, p_2 ∈ P} ∪
{(p, has_agent, a) ∈ SS | p ∈ P ∧ a ∈ A} ∪
{(p, has_participant, d) ∈ SS | p ∈ P ∧ d ∈ D} ∪
{(p_1, part_of, p_2) ∈ SS | p_1, p_2 ∈ P} ∪
{(a, has_parameter, pa) ∈ SS | a ∈ A ∧ pa ∈ D} ∪
{(a_1, adjacent_to, a_2) ∈ SS | a_1, a_2 ∈A} ∪
{(a_1, part_of, a_2) ∈ SS | a_1, a_2 ∈A} ∪
{(a_1, contained_in, a_2) ∈ SS | a_1, a_2 ∈A} ∪
{(d_1, part_of, d_2) ∈ SS | d_1, d_2 ∈D} ∪
{(d_1, contained_in, d_2) ∈ SS | d_1, d_2 ∈D} ∪
{(d_1, transformation_of, d_2) ∈ SS | d_1, d_2 ∈D} ∪
{(d_1, derives_from, d_2) ∈ SS | d_1, d_2 ∈D}
Note: *The definition of the set P is recursive and it is checked that the set P is well-defined.*

The *provenance ( )* query operator is expensive to implement and is not required in most scenarios. For example, provenance metadata generated by a scientific workflow represents a subset of the results returned by this operator. Hence, we define a specialized form of this operator with additional constraints called *provenance_pathway ( )* to retrieve workflow-specific provenance metadata.

1. ***provenance_pathway ( )***: This operator returns provenance information of a data entity consisting of only an individual belonging to `data_collection` and `process` classes. This operator reflects requirements of workflow provenance, where provenance consists of a series of "activities" (where an "activity" is any sub-class of `process`) and set of "data entities" (where a "data entity" is any subclass of `data_collection` class) that form the input and output of the processes. A set of constraints defined as an expression over individuals belonging to `data_collection`, `agent`, `process`, and `parameter` can also be defined as part of the input. For example, query involving

workflow provenance of ocean current values may have a time interval constraint as April 21, 2003 to May 2, 2003.

Definition 2: *provenance_pathway ()*

SS: Search Space (set of all available triples)
Proc = {p | (p, rdf:type, process) ∈ SS}
DataC = {d | (d, rdf:type, data_collection) ∈ SS}
Input: dc

IN = {t ∈ SS | t = (s, has_participant, dc)}

P is smallest set such that,
P = ({p | ∃ x, y. (P, x, y) ∈ IN} ∪ {p | ∃ x ∈ P. (x, preceded_by, p)}) ∩ Proc
D = {d | (p, has_participant, d) ∈ SS ∧ p ∈ P} ∩ DataC
OUT = IN ∪ {(p_1, preceded_by, p_2) ∈ SS | p_1, p_2 ∈ P} ∪
{(p, has_participant, d) ∈ SS | p ∈ P ∧ d ∈ D} ∪
{(p_1, part_of, p_2) ∈ SS | p_1, p_2 ∈ P} ∪
{(d_1, part_of, d_2) ∈ SS | d_1, d_2 ∈D} ∪
{(d_1, contained_in, d_2) ∈ SS | d_1, d_2 ∈D} ∪
{(d_1, transformation_of, d_2) ∈ SS | d_1, d_2 ∈D} ∪
{(d_1, derives_from, d_2) ∈ SS | d_1, d_2 ∈D}

Note: *The definition of the set P is recursive and it is checked that the set P is well-defined.*

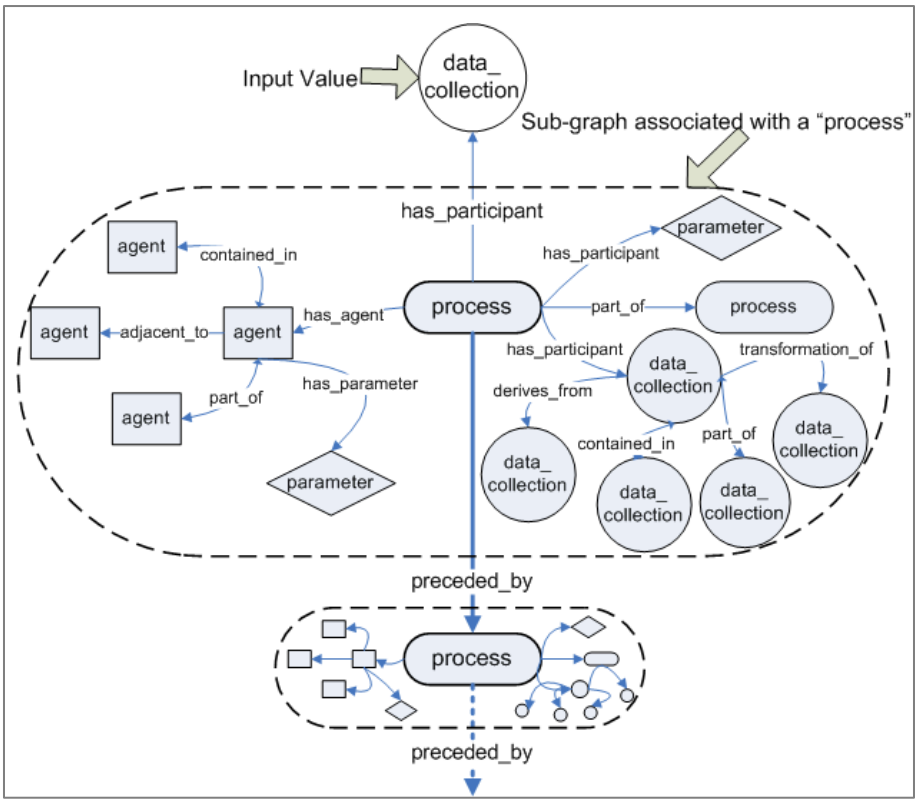


Figure 4-1 Schematic representation of *provenance ()* query operator execution strategy

Next, we define the ***provenance_context*** query operator.

4.3.3 *provenance_context ( )* Query Operator

This query operator supports the provenance queries in **C*ategory II***. The query operator takes as input provenance values as constraints and returns data entities that satisfy the provenance constraints. In effect, the query input values define a formal "provenance context" composed of constraint values defined over all available provenance information. For example, the provenance details such as ocean buoy identifier- "*oceanBuoy7044*", geographical coordinates- "*475111N:1222118W*", and temporal constraints-"*April 21, 2003 to May 2, 2003*" form a very specific contextual structure that is used to identify data entities that satisfy these provenance constraints. The data entities in the query result have similar or equivalent provenance, and hence, can be interpreted with equal level of trust. The query operator can be formally defined using the *provenance ()* query operator.

Definition 3: *provenance_context ()*

SS: Search Space (set of all available triples)
DataC = {d | (d, rdf:type, data) ∈ SS}
Input: set of triples pc = $pc_g \cup pc_v$,
where pc_g describes provenance values directly connected to data individuals and pc_v includes variables in input triples.

Output = {dc | ($pc_g \subseteq$ provenance (dc)} ∩
{dc | for all (v, x, y) ∈ pc_v we have (dc, x, y) ∈ SS} ∩
{dc | for all (x, y, v) ∈ pc_v we have (x, y, dc) ∈ SS} ∩
{dc | (dc, rdf:type, DataC) ∈ SS}

Similar to the ***provenance_context ( )*** operator that returns individuals of class `data_collection`, specialized operators can be defined for individuals of `process` and `agent` classes.

1. ***pc_process ()***: For a given set of constraints, retrieve all the instances of `process` class that satisfy the set of constraints. An example query from the oceanography scenario is "*Find all invocations of a computational process "HyperCubetoDataTable" with parameter "InverseData" set to value = false*".

Definition 4: *pc_process ()*

SS: Search Space (set of all available triples)
ProcessC = {d | (d, rdf:type, process) ∈ SS}
Input: set of triples $pc = pc_g \cup pc_v$,
where pc_g describes provenance values directly connected to data individuals and pc_v includes variables in input triples.

Output = {dc | ($pc_g \subseteq$ provenance (dc)} ∩
{dc | for all (v, x, y) ∈ pc_v we have (dc, x, y) ∈ SS} ∩
{dc | for all (x, y, v) ∈ pc_v we have (x, y, dc) ∈ SS} ∩
{dc | (dc, rdf:type, ProcessC) ∈ SS}

2. ***pc_agent ()***: For given a set of constraints retrieve all the instances of `agent` class that satisfy the constraints.

Definition 5: *pc_agent ()*

SS: Search Space (set of all available triples)
AgentC = {d | (d, rdf:type, agent) ∈ SS}
Input: set of triples $pc = pc_g \cup pc_v$,
where pc_g describes provenance values directly connected to data individuals and pc_v includes variables in input triples.

Output = {dc | ($pc_g \subseteq$ provenance (dc)} ∩
{dc | for all (v, x, y) ∈ pc_v we have (dc, x, y) ∈ SS} ∩
{dc | for all (x, y, v) ∈ pc_v we have (x, y, dc) ∈ SS} ∩
{dc | (dc, rdf:type, AgentC) ∈ SS}

4.3.4 *provenance_compare ( )* Query Operator

Accurate comparison of scientific results requires the comparison of the associated provenance information. For example, two ocean visualization charts are said to be *comparable* if the associated provenance information (type of sensors, parameters used in the scientific workflow) are identical. We use the RDF graph equivalence definition [107] with the added functionality of "coloring" the nodes and labeling the edges using the Provenir ontology schema, to define equivalence between two provenance graphs. The formal definition of the query operator is as follows:

Definition 6: *provenance_compare ()*

Input: G_1, G_2
$G_1 = (V_1, E_1, l_1, m_1)$
$G_2 = (V_2, E_2, l_2, m_2)$
The two provenance graphs G_1 and G_2 are equivalent if there are bijections,
$i = V_1 \rightarrow V_2$ and
$j = E_1 \rightarrow E_2$ such that
$\forall (e_1, e_2) \in E_1$ we have $(i(e_1), i(e_2)) \in j(e_1, e_2)$ and
$\forall x \in V_1 : m_1(x) = m_2(i(x))$ holds.

The next query operator enables the merging of provenance information. For example, provenance from different stages of an experiment protocol can be merged to construct an unified view of the experiment process.

4.3.5 *provenance_merge ( )* Query Operator

The query operator takes as input two provenance graphs and gives as output a single, merged provenance graph. The merged graph does not include any duplicates. A formal definition of the merge operator is given below.

Definition 7: *provenance_merge ()*

Input: G_1, G_2
$G_1 = (V_1, E_1, l_1, m_1)$
$G_2 = (V_2, E_2, l_2, m_2)$
Output: $(V_1 \cup V_2, E_1 \cup E_2, l, m)$, where
$l(x) = \{ l_1(x)$ if $x \in V_1 \cup E_1$
$l_2(x)$ if $x \in V_2 \cup E_2$ and where
$m: V_1 \cup V_2 \rightarrow Nc : m(x) = \{ m_1(x)$ if $x \in V_1, m_2(x)$ if $x \in V_2$

In the next section, we correlate the provenance query operators to existing work.

4.4 Related Work

Provenance has been studied from multiple perspectives across a number of domains. In the next section, we correlate the provenance query operators introduced in this paper to existing work in database and workflow provenance as illustrated in Figure 4-2.

4.4.1 Database Provenance

Database provenance or data provenance, often termed as "fine-grained" provenance [58], has been extensively studied in the database community. Early work includes the use of annotations to associate "data source" and "intermediate source" with data (polygen model) in a federated database environment

to resolve conflicts [108], and use of "attribution" for data extracted from Web pages [109]. More recent work has defined database provenance in terms of "Why provenance", "Where provenance" [36], and "How provenance" [37].

A restricted view of the "Where provenance" identifies each piece of input data that contributes to a given element of the result set returned by each database query. "Why provenance" was first described in [35]. We use the syntactic definition of "Why provenance" [36] that defines a "proof" for a data entity. The proof consists of a query, representing a set of constraints, over a data source with "witness" values that result in a particular data output. The semantics of the *provenance ()* query operator closely relates to both "Where provenance" and "Why provenance" [36]. To address the limitation of "Why provenance" that includes "…set of all contributing input tuples" leading to ambiguous provenance, [37] introduced semiring-based "How provenance." The *provenance ()* query operator over a "weighted" provenance model, which reflects the individual contribution of each component (for example process loops or repeated use of single source data), is comparable to "How provenance."

The Trio project [96] considers three aspects of lineage information of a given tuple, namely, how was a tuple in the database derived along with a time value (when) and the data sources used. A subset of queries in Trio, "lineage queries", discussed in [96], can be mapped both as *provenance ()* and as *provenance_context ()* query operators depending on the input value. The SPIDER system [110] built on top of Clio [111] uses provenance information modeled as "routes" (schema mappings) between source and target data, to capture aspects of both "Why provenance" and "How provenance". Hence, it closely relates to the semantics of the *provenance ()* query operator.

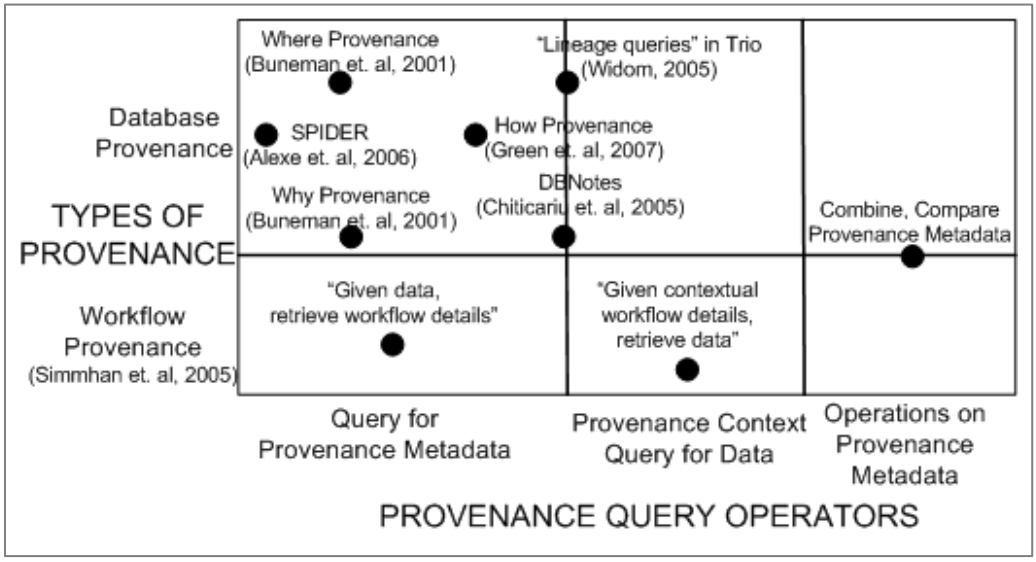


Figure 4-2 Mapping provenance query operators with existing database and workflow provenance

4.4.2 Workflow Provenance

The rapid adoption of scientific workflows to automate scientific processes has catalyzed a large body of work in recording provenance information for the generated results. Simmhan et al. [38] survey different approaches for collection, representation and management of workflow provenance. The participants in the provenance challenge [39] collected provenance at different levels of granularities such as comprehensive workflow system traces in PASS [112], use of semantic annotations of services by Taverna [113], and recording of data value details and service invocations in Karma [87]. Recent work has also recognized the need for inclusion of domain semantics in the form of domain-specific provenance metadata [5] along with workflow provenance. The semantics of these projects can be mapped to the *provenance ()* query operator.

4.5 Conclusions

In this chapter, we discussed the foundations of a provenance query infrastructure by (1) analyzing the characteristics of provenance queries, (2) introducing a classification scheme for provenance queries, and (3) defining a set of provenance query operators. The provenance classification scheme identifies four categories of provenance queries. Using the classification scheme, we described a set of provenance query operators that use the Provenir ontology as the reference model (corresponding to a TBox) and are executed over provenance RDF graphs (corresponding to the ABox). This allows the provenance query operators to be used in multiple application domains that use the Provenir ontology to represent provenance information.

Chapter 5

Implementing the Provenance Query Infrastructure

Introduction

In the previous chapter, we introduced a set of provenance query operators as a step toward creating a provenance query infrastructure. The primary objective of the implementation is to create an infrastructure that can be easily added to existing scientific applications using Semantic Web technologies and is reusable across multiple domains. Hence, completing the use of the Provenir ontology schema (expressed in OWL [40]), the RDF representation model to store provenance information, we represent the provenance queries using the SPARQL RDF query language [69]. In this chapter, we discuss (a) the implementation of the query operators in a provenance query engine, (b) analyze the complexity of provenance query patterns in the SPARQL RDF query language that makes a naïve implementation of the provenance query engine impractical for use in real world applications, and (c) the use of a novel materialization strategy that leverages the Provenir ontology to optimize provenance queries. We first describe the architecture of the provenance query engine.

5.2 Provenance Query Engine

The provenance query engine is designed as a Java-based Application Programming Interface (API) to support the provenance query operators over a RDF data store. The query engine described in this section is integrated with an Oracle 10g (release 10.2.0.3.0) data store, but we note that the query engine can be used with any RDF data store that supports SPARQL [69] and inference rules. The Oracle 10g RDF data store uses a SQL table function (RDF_MATCH) to efficiently query RDF data [114]. The default Oracle 10g query interface does not support all SPARQL functions, hence we used the Oracle's Jena [62] based plug-in, which supports the full SPARQL specification.

The provenance query engine consists of three functional components (Figure 5-1):

1. **A Query Composer**: The query composer maps the provenance query operators to SPARQL syntax according to semantics of the query operators.
2. **A Function to Compute Transitive Closure over RDF**: SPARQL query language does not support transitive closure for an RDF <*node, edge*> combination. Hence we have implemented a function to efficiently compute transitive closure using the SPARQL ASK function. The result of this function is used by the query composer.

3. **Query Optimizer using Materialized Provenance Views**: Using a new class of materialized views based on the Provenir ontology schema called Materialized Provenance Views (MPV) a query optimizer has been implemented that enables the query engine to scale with very large RDF data sets.

We now describe the SPARQL query composer and transitive closure function. First, we briefly describe the SPARQL language structure and the approach used by the query composer to map the provenance query operator to SPARQL. A SPARQL query is composed of a set of triples (each triple is of the form, <*Subject*, *Property*, *Object*>) to form a query pattern called basic graph pattern (BGP), where any one of the three constituents may be a variable. For example, <*?x*, *is_president_of*, *UnitedStateofAmerica*> is a triple pattern with the variable '*?x*'. The BGP is used to find a sub-graph from the RDF store where the values in the sub-graph may be substituted for the variables in the BGP resulting in a RDF graph equivalent to the sub-graph [69]. But, if suitable instantiation of variables in BGP are not found in the RDF store, no results are returned [69].

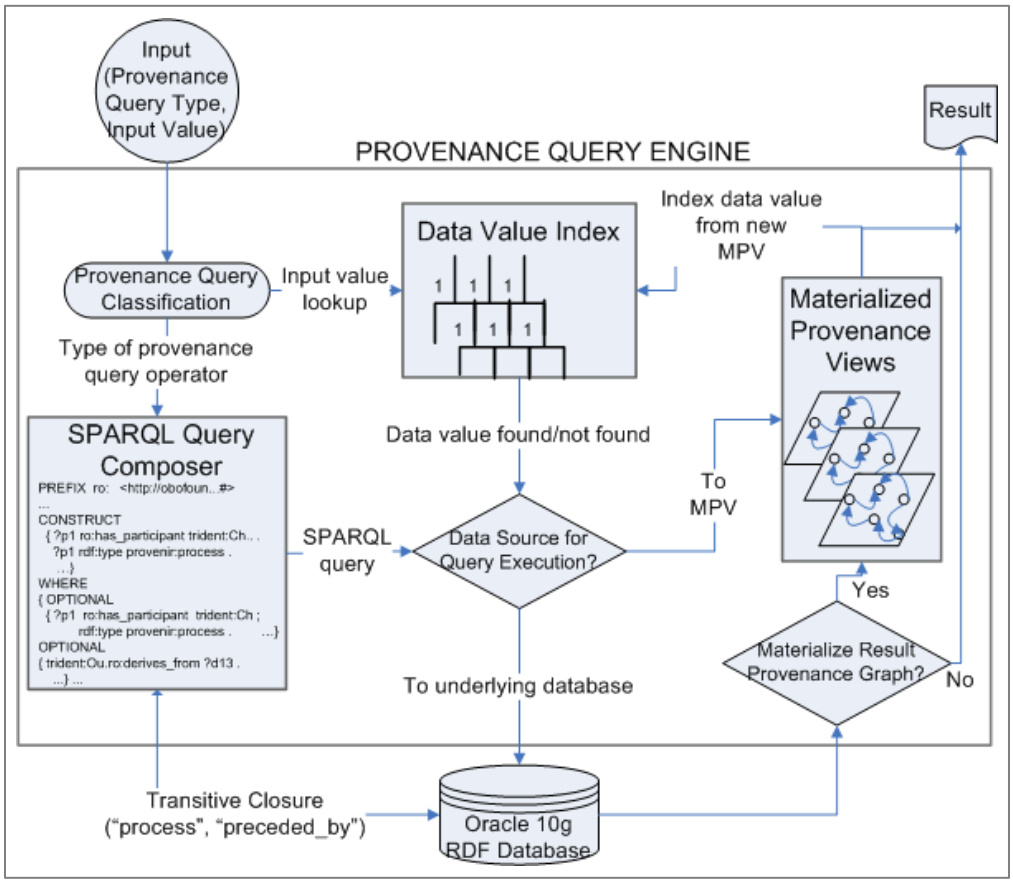


Figure 5-1 Architecture of the provenance query engine

The provenance query operators represent the complete result set by defining exhaustive set of dependencies among data, process, and agent. However, in real world scenarios the provenance

information available can be incomplete due to application-specific or cost-based limitations. Hence, a straightforward mapping of provenance query operators to SPARQL as a BGP is not desirable. Such a BGP-based query expression pattern may not return a result in the presence of incomplete provenance information. The OPTIONAL function in SPARQL can be used to specify query expression patterns that can succeed with partial instantiation, yielding maximal "best match" result graph. Hence, the query composer uses this OPTIONAL function to create a query expression pattern.

Further, as discussed in Section 5.1.1, the query composer also needs to compute the transitive closure over the <process, preceded_by> combination to retrieve all individuals of the process class linked to the input value. But, unlike many graph database query languages such as Lorel or GraphLog, [115], SPARQL does not provide an explicit function for transitive closure to answer reachability queries.[2] We now describe the transitive closure function implemented in the provenance query engine.

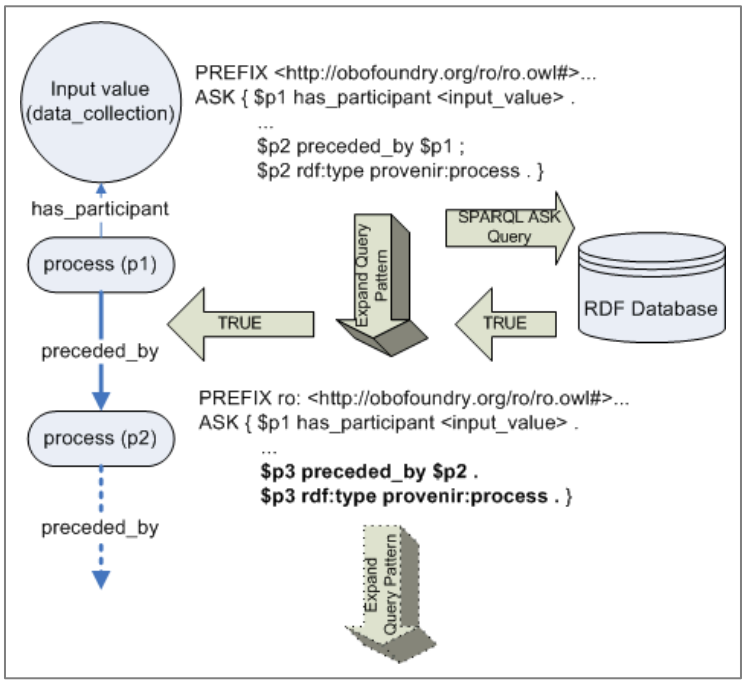


Figure 5-2 RDF transitive closure using SPARQL ASK function

5.2.1 Computing Transitive Closure

We had two options in implementing the transitive closure function, namely a function that is tightly couple to the RDF store or a generic function. We chose a generic implementation using the SPARQL ASK function that allows the provenance query engine to be used over multiple RDF stores. The

[2]The W3C DAWG postponed a decision on transitive closure in SPARQL

SPARQL ASK function allows "application to test whether or not a query pattern has a solution," [69] without returning a result set or graph. The transitive closure function starts with the process instance (p1) linked to the input value and then recursively expands the SPARQL query expression using the ASK function till a *false* value is returned, thereby terminating the function (Figure 5-2). The SPARQL ASK function, in contrast to the SELECT and CONSTRUCT functions, does not bind the results of the query to variables in the query pattern. Hence, it is a low-overhead function for computing transitive closure. The results of a comparative evaluation of the SPARQL functions along with the performance evaluation of an implementation of the provenance query engine are presented in the next section.

5.3 Evaluation Strategy

The objectives of evaluating the provenance query engine was three fold:

1. Evaluate the functionality of the provenance query operators in terms of their ability to accurately answer the provenance queries,
2. Evaluate the scalability of the provenance query engine with increasing size of data and complexity of the query patterns, and
3. Comparison of the transitive closure function using SPARQL ASK, SELECT, and CONSTRUCT functions.

The data and queries used in the evaluation are from the Neptune oceanography project.

5.3.1 The Neptune Oceanography Project

The Neptune project [104], led by the University of Washington, is an ongoing initiative to create a network of instruments widely distributed across, above, and below the seafloor in the northeast Pacific Ocean. We consider a simulated scenario, illustrated in Figure 5-3, involving data collection by ocean buoys (containing a temperature sensor and an ocean current sensor), and data processing by a scientific workflow that creates visualization charts as output.

We consider two scenarios that require the use of provenance information for analyzing and managing the data in this project:

1. One of the ocean buoys is found to be damaged due to a recent storm. Hence, all visualization charts created using data from this ocean buoy, after it was damaged, need to be discarded and new charts should be created using correct sensor data. This is a typical provenance related query and can be addressed using two approaches. In the first approach, the provenance information of each visualization chart can be analyzed to locate the relevant set of possibly inaccurate visualization charts. In the second approach, a set of constraints can be defined on the provenance information to

retrieve the data entities that satisfy these constraints, for example "given the identifier of the damaged ocean buoy, locate all visualization charts that used data from this sensor" (generated after the ocean buoy was damaged).

2. Another typical scenario involves the comparison of provenance information associated with given results. For example, oceanography researchers can query for two visualization charts that were generated from sensor data collected under equivalent weather conditions. The user can define the context for comparison of the provenance information in terms of wind speed, temperature, and precipitation.

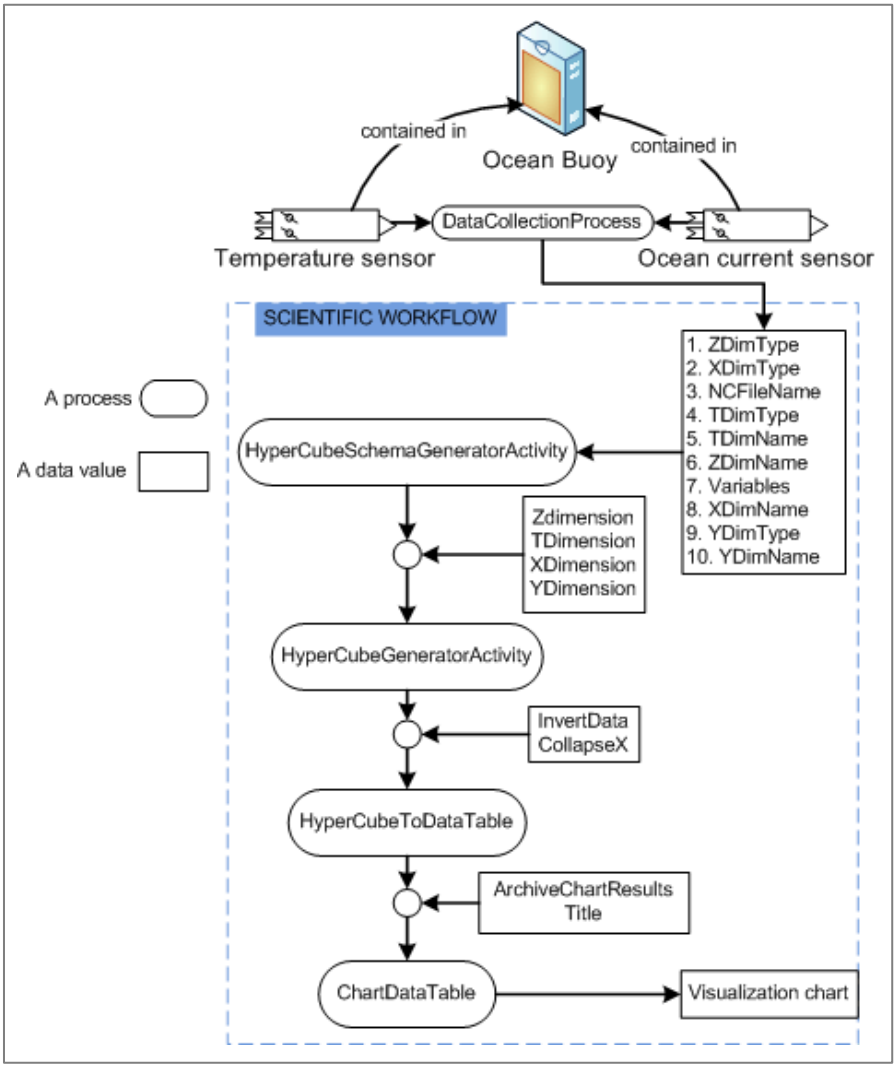


Figure 5-3 A simulated oceanography scenario from Neptune Project

We use the data and the associated provenance information from this scenario to evaluate the performance of the provenance query engine.

5.3.2 Evaluating the Data and Expression Complexity of Provenance Query Operators

We use the two standard complexity measures of (a) **Query or Expression Complexity** (varying the syntax of the query with fixed data size), and (b) **Data Complexity** (vary the data size with a fixed query expression) [116], to characterize the performance of the provenance query engine. The evaluation results presented in this section are for the ***provenance ()*** query operator, which represents the majority of provenance queries.

The expression complexity of the SPARQL graph pattern using the OPTIONAL function is PSPACE-complete [117]. As discussed earlier in this section, the SPARQL query pattern for the *provenance ()* query operator requires the use of OPTIONAL function with multiple levels of nesting. The other components that affect the evaluation of a SPARQL query are the total number of variables and the number of triples defined in the query pattern. Hence, to evaluate the expression complexity of the *provenance ()* query operator, we use five queries (listed in Table 5.1) with increasing number of variables, triple patterns, and nesting levels using the OPTIONAL function over a fixed dataset size. The five queries, from the Neptune oceanography project, involve retrieval of provenance information associated with different data entities. The full SPARQL query pattern corresponding to Query 1 and Query 5 (in Table 5.1) are listed in Appendix A.

Table 5.1: Queries (expressed in SPARQL syntax) used to evaluate expression complexity of the *provenance ( )* query operator

Query: Retrieve Provenance of given Input Value	**Number of Variables**	**Number of Triples**	**Nesting using OPTIONAL**
Q1. codar_mnty_908294932772185.nc	31	86	4 levels
Q2. NetCDFReader90829493474170	45	126	5 levels
Q3. HyperCubeSchema 90829493462995	45	126	5 levels
Q4. HyperCube 90829493567757	59	166	6 levels
Q5. ChartDataTable 90829493849637	73	206	7 levels

5.3.3 Experiment Setup and Dataset

The experiments were conducted using Oracle10g (Release 10.2.0.3.0) DBMS on a Sun Fire V490 server running 64-bit Solaris 9 with four 1.8 GHz Ultra Sparc IV processors and 8GB of main memory. The database used an 8 KB block size and was configured with a 512 MB buffer cache. The timings reported are mean result from five runs with warm cache.

The dataset for the evaluations was generated from the oceanography scenario described in Section 5.2.1. The scientific workflow was executed using the Trident workflow workbench [118]. The Trident log file with additional details such as temperature sensors, ocean current sensors, and ocean buoys, was used as a template to generate the RDF data.

Five datasets corresponding to 100, 1000, 10000, 100000, and 1 million experiment cycles are used (Table 5.2). The largest dataset corresponding to 1 million experiment cycles contains about 308 million RDF triples. A rule index [114] was defined for inferencing over the datasets using standard RDFS rules [25] and a user-defined rule to infer that, "If the input value of a process (p1) is the same as the output value of another process (p2), then p1 is linked to p2 by the property `preceded_by`". Table 5.2 lists the total number of new RDF triples inferred using the RDFS and user-defined rules for each of the five datasets.

Table 5.2: RDF datasets to evaluate the data complexity using the *provenance ( )* query operator

Dataset Id	Number of Experiment Cycles	Number of Inferred RDF Triples	Total Number of RDF Triples
DS1	100	23,620	32,182
DS2	1000	226,671	309,459
DS3	10,000	2,257,096	3,082,184
DS4	100,000	22,560,776	30,808,427
DS5	1,000,000	225,596, 929	308, 069,953

5.4 Results

Figure 5-4 illustrates the individual contribution of three components of the query engine, transitive closure, query composer and query executions modules to the total query time for increasing complexity of SPARQL query patterns (Table 5.1). We note that as the expression complexity increases the query execution time dominates the total query time as compared to the time taken for computing the transitive closure and query composer.

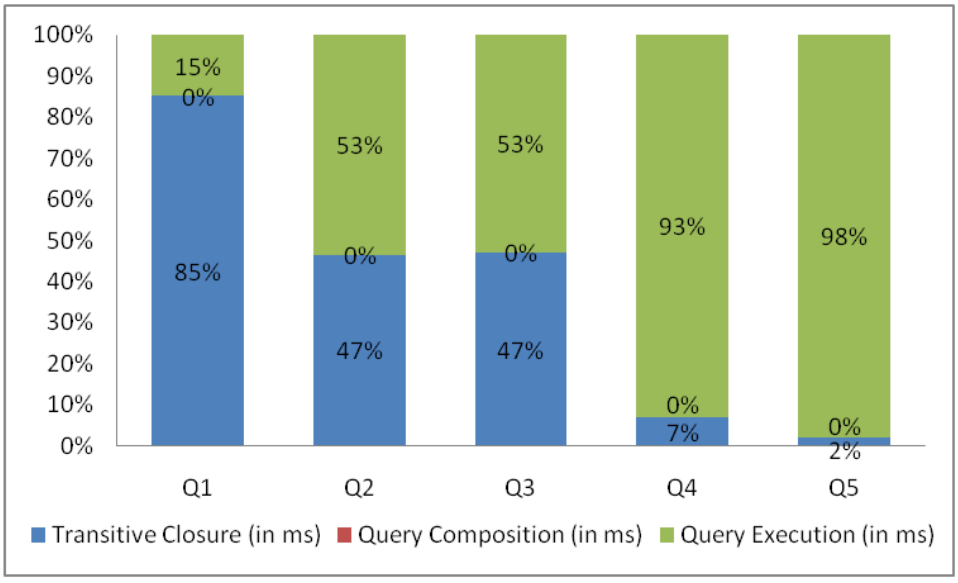


Figure 5-4 Contribution of each provenance query engine component (in %)

We now discuss the results comparing the use of the three SPARQL functions - ASK, SELECT, and CONSTRUCT, followed by the query performance of an implementation of the provenance query engine.

5.4.1 Transitive Closure Computation using SPARQL Functions

This experiment compares the performance of transitive closure computation using the SPARQL "SELECT", "CONSTRUCT", and "ASK" functions. The transitive closure for the five queries, Q1 to Q5 (in Table 5.1), is computed using the largest dataset DS5 (in Table 5.2) of 308 million RDF triples.

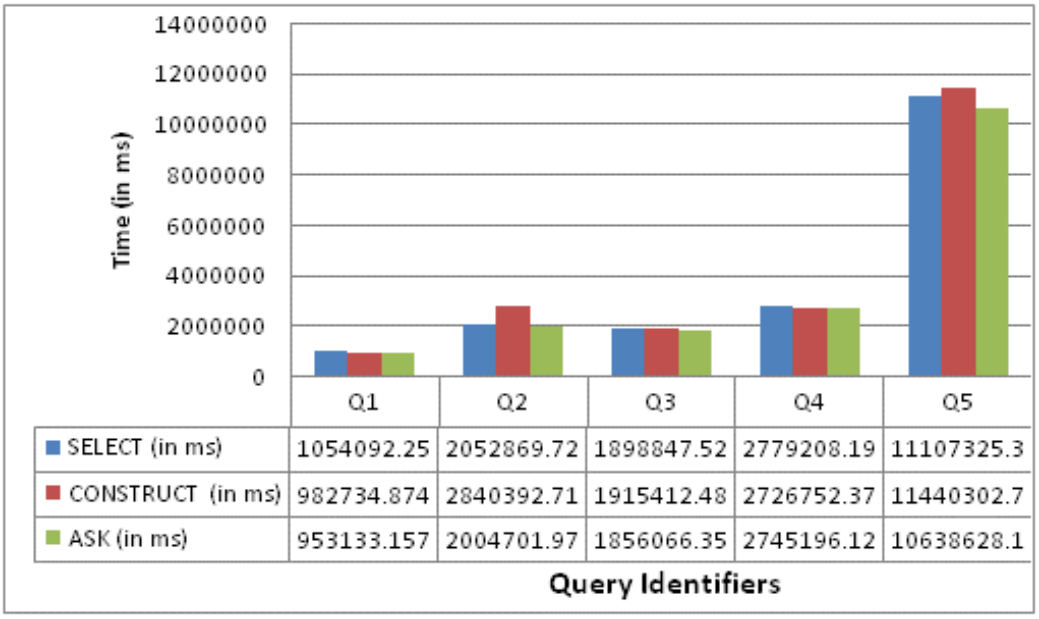


Figure 5-5 Performance results for the three SPARQL functions to compute transitive closure on <process, preceded_by>

The results in Figure 5-5 show that the ASK function consistently performs better than both the SELECT and the CONSTRUCT functions. This is because both SELECT and CONSTRUCT involve a binding of query results to query pattern variables and graph pattern respectively. In contrast, the ASK function returns a Boolean value indicating if a graph corresponding to the query pattern exists in the RDF store. Hence, the ASK-function based transitive closure function implemented in the query engine is a low-overhead solution.

5.4.2 Query Expression Complexity

This experiment characterizes the expression complexity of the *provenance ( )* query operator in SPARQL syntax. The results are for the total query time, including time for transitive closure, query composition, and query execution, for the five provenance queries, Q1 to Q5 (in Table 5.1). The queries are executed against a fixed size dataset, DS5 (in Table 5.2), with about 308 million RDF triples representing 1 million experiment cycles. The results (Figure 5-6) demonstrate that the query time

increases with increasing expression complexity of the queries. Thus a straightforward implementation of the query engine is unusable for provenance management systems in eScience projects.

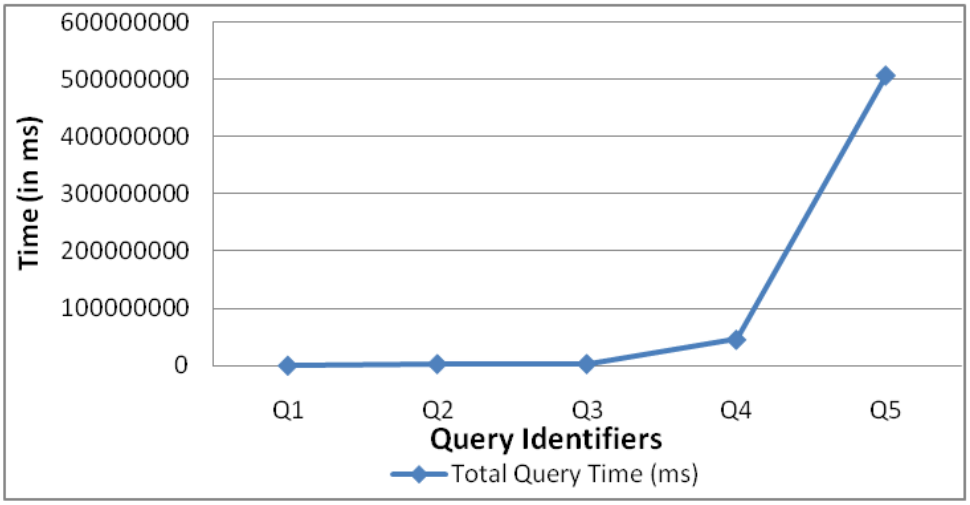


Figure 5-6 Evaluation results for expression complexity of provenance queries illustrating the significant increase in query time with increasing expression complexity

5.4.3 Data Complexity

This experiment characterizes the data complexity of the *provenance ( )* query operator in SPARQL syntax. The query Q5 (in Table 5.1) with maximum expression complexity is used as the fixed query for evaluation over varying sizes of RDF datasets (in Table 5.2). Figure 5-7 illustrates that the data complexity of the *provenance ()* query operator is also large and an effective optimization technique is necessary for use of the provenance query engine in a practical provenance management system.

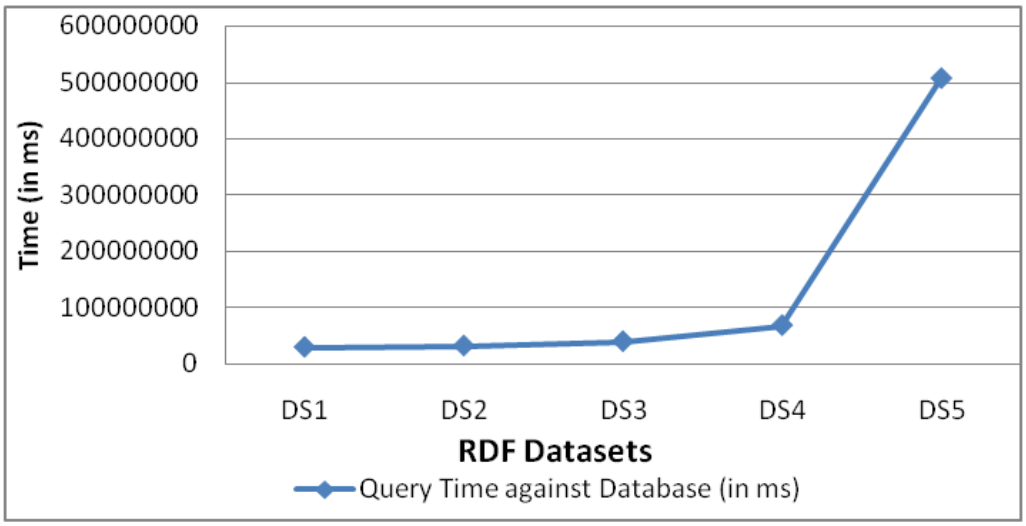


Figure 5-7 Evaluation results for data complexity of provenance queries illustrating the significant increase in query time with increasing data complexity

Overall, the evaluation results clearly demonstrate that a straightforward implementation of the provenance query engine cannot scale with both increasing complexity of provenance query expressions and size of provenance data. In the next section, we introduce a new class of materialized views for query optimization to address both these issues.

5.5 Materialized Provenance Views: Optimizing Provenance Queries

The provenance queries are graph traversal operations for path computations. Path computation over graphs is an expensive operation especially in the case of provenance queries that require computation of fixed paths, recursive pattern-based paths and neighborhood retrieval. The results of our evaluation show that even industrial strength RDF database face severe limitations in terms of response time for complex provenance queries over large scientific datasets. To address this we defined a new class of materialized views called materialized provenance views (MPV) that materializes provenance sub-graphs for selected classes of input values. We considered two constraints to decide what data should be materialized, (a) the cost of maintaining the materialized views, that is, if a materialized view needs to be recalculated when new RDF triples are added to the database, and (b) the number and complexity of provenance queries that can be satisfied by a MPV.

Provenance metadata by definition describes past events and therefore they are not subject to frequent updates, except in the presence of errors. Therefore materialized views are a suitable approach for provenance query optimization. For the second constraint, we use the Provenir model to identify one logical unit of provenance information that can be used to satisfy not one but multiple provenance queries. The provenance graph of the final output of an experiment cycle, the "Chart Visualization" file (Figure 5-8) represents a logical unit of provenance information for the Neptune oceanography project scenario. This provenance query engine computes the unit of provenance information using the Provenir ontology.

The MPV (Figure 5-8) can not only satisfy the provenance query for "*Chart Visualization*" files, but also provenance queries for all entities occurring in that one experiment cycle. A B-tree index is used to index all instances of the data_collection class that occur in that experiment cycle. Given an input value, the query optimizer looks-up the B-tree index to decide if the query can be satisfied by a MPV or by the underlying database. MPVs are created using a memoization approach instead of an eager strategy, since the total sum of MPVs created for all final output values equals the database itself. In the next section we discuss the evaluation results using MPV for query optimization.

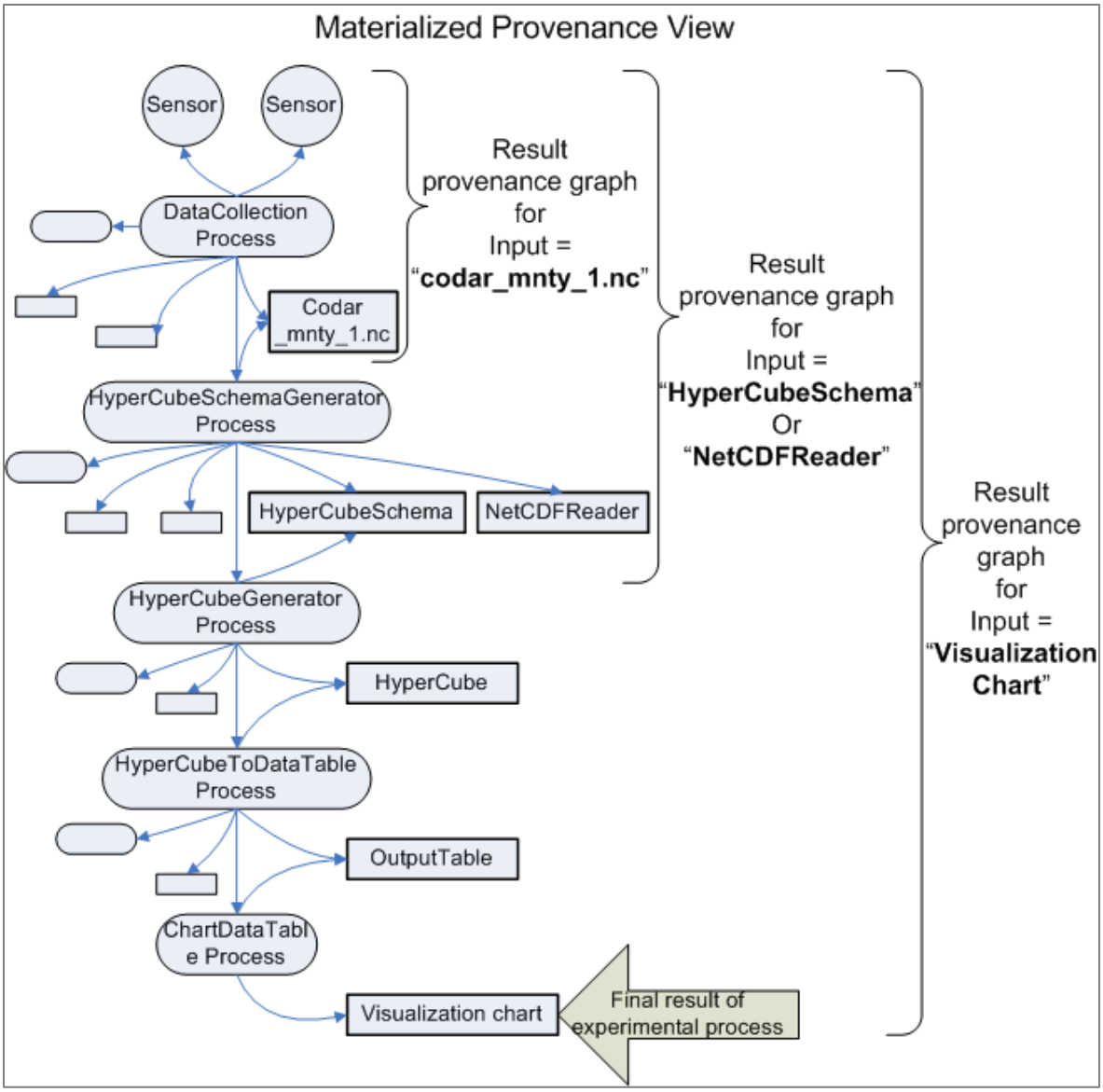


Figure 5-8 A MPV corresponds to one logical unit of provenance in the given application domain

5.5.1 Query Expression Complexity using MPV for Query Optimization

We use a straightforward implementation (discussed earlier in Section 5.3.2 and Section 5.3.3) as benchmark to discuss the performance of the provenance query engine using MPV. In the first experiment, queries from Section 5.3.2 are evaluated against a MPV. The MPV used in this experiment corresponds to one experiment cycle with final output "*ChartDataTable90829493849637*"; the MPV contained 86 RDF triples and occupied 12KB.

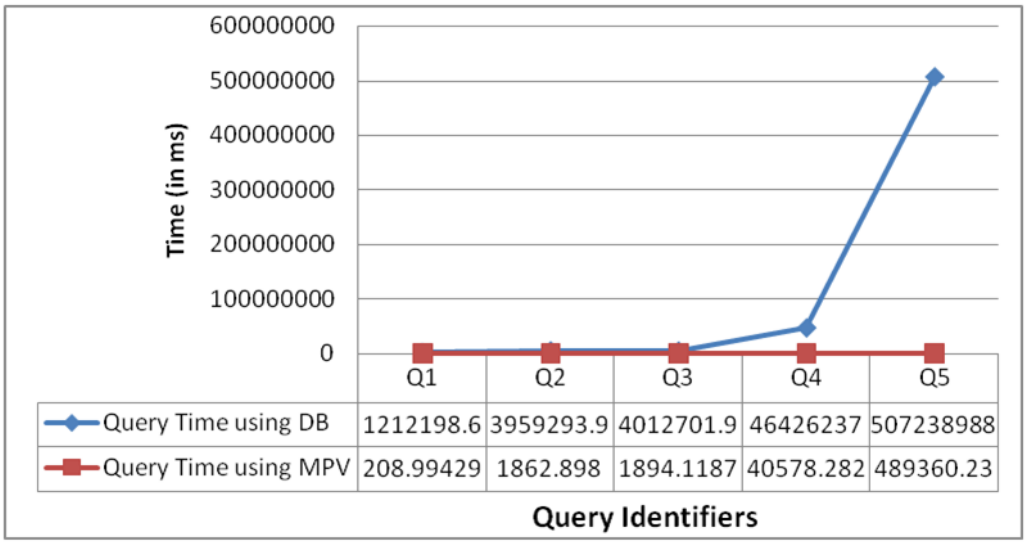


Figure 5-9 Comparing expression complexity results with and without using MPV

Figure 5-9 clearly demonstrates the significant reduction in total query time through the use of MPV. Note that a single MPV is used to satisfy all provenance queries from Table 5.1. The percent gain in performance with MPV as compared to query execution against the database is listed in Table 5.3.

5.5.2 Data Complexity using MPV for Query Optimization

Similarly, the results in Figure 5-10 show significant speed-up in query time over varying sizes of RDF datasets using the fixed query, Q5 (Table 5.1). Table 5.3 lists the performance improvement for data complexity using MPV for query optimization.

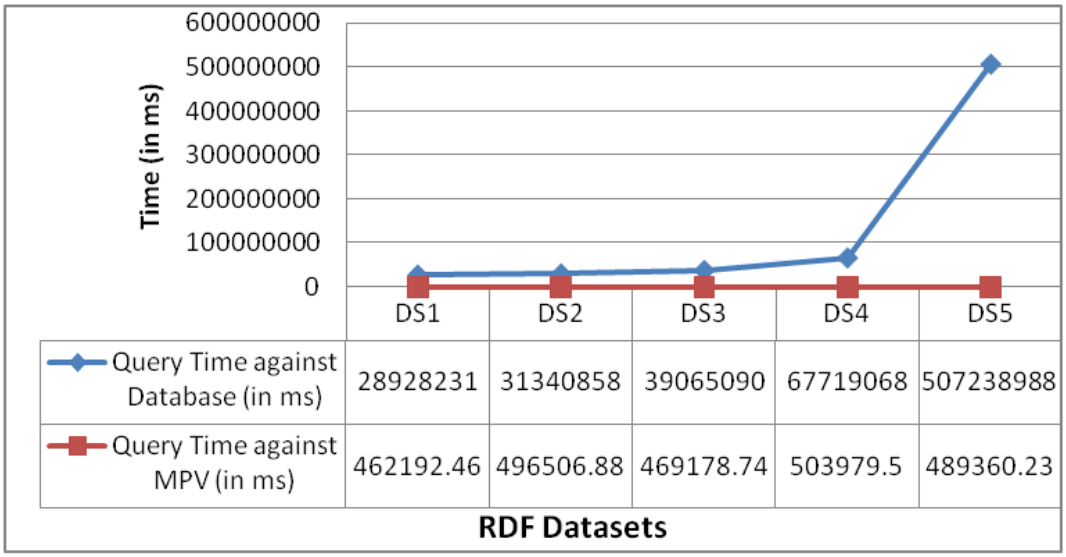


Figure 5-10 Comparing data complexity results with and without using MPV

Table 5.3 Performance gain for *provenance ( )* query operator using MPV

For Expression Complexity (gain in %)		For Data Complexity (gain in %)	
Q1	99.98	DS1	98.40
Q2	99.95	DS2	98.42
Q3	99.95	DS3	98.80
Q4	99.91	DS4	99.26
Q5	99.90	DS5	99.90

5.6 Related Work

The query language for provenance (QLP) [98] is roughly comparable to the functionalities of the provenance query engine discussed in this chapter. The QLP defines query constructs that allows easier composition of provenance queries. Unlike the RDF representation model supported by the provenance query engine, QLP assumes a XML-based provenance representation model and is primarily focused on workflow provenance. The vtPQL, used in the VisTrails project [119], also defines query constructs for provenance queries, but similar to QLP vtPQL is primarily focused on workflow provenance [120].

5.7 Conclusions

In this chapter, we discussed the implementation of the provenance query operators in a provenance query engine. The provenance query engine consists of three components namely, (a) a query composer, (b) a function to efficiently compute transitive closure over `<process, preceded_by>` node-edge combination, and (c) a novel materialized view based optimization technique that uses the Provenir ontology schema to identify the specific RDF sub-graph to be materialized. The MPV represents a logical unit of provenance and is computed automatically for different domains by the provenance query engine using the standard RDFS entailment rules. The provenance query engine is a practical application that can be easily added to existing scientific applications to support complex provenance queries over increasing sizes of datasets.

Chapter 6

Provenance Context Entity (PaCE): Scalable Provenance Tracking for Scientific RDF Data

Introduction

An increasing number of scientific applications are storing and disseminating their datasets using the Semantic Web Resource Description Framework (RDF) format [29] [85] [4]. RDF is also being used as an information integration platform in multiple scientific domains. The Biomedical Knowledge Repository (BKR) project at the U.S. National Library of Medicine is creating a comprehensive repository of integrated biomedical data from a variety of sources such as biomedical literature (textbooks and journal articles), structured data bases (for example the NCBI Entrez system [32]), and terminological knowledge sources (for example, the Unified Medical Language System (UMLS) [121]) [122]. BKR represents the integrated information in RDF, for example, the RDF statement "`lipoprotein→affects→inflammatory_cells`" was extracted by a text mining tool from a journal article (with PubMed identifier PMID: 17209178) and states that `lipoprotein` (denoted as "subject" of the RDF triple3) `affects` (denoted as "property" of the triple) `inflammatory_cells` (denoted as the "object" of the triple). In addition to the advantages of more expressive modeling [123], storing information as RDF statements enables BKR to be compatible with the rapidly growing Linked Open Data (LOD) initiative that currently has more than 4.2 billion RDF statements representing a large number of domains including biomedicine, census data, chemistry, and geography [23].

In addition to the biomedical data, BKR also records and uses provenance metadata describing the history or lineage of the RDF statements. The provenance information identifies the source of an extracted RDF triple, temporal information (for example, the date of publication of a source article), version information for a database, and the confidence value associated with a triple (indicated by a text mining tool). The provenance information is essential in the BKR project to ensure the quality of data and associate trust value with the RDF triple. It has specific applications in the four services offered by the BKR namely, enhanced information retrieval (search based on the named relationship linking two entities), multi document summarization, question answering, and knowledge discovery. We describe example scenarios that highlight the use of provenance information in the four services offered by BKR:

[3] We use the notions *RDF statement*, *RDF triple*, and *triple* interchangeably.

1) *Enhanced information retrieval service*: Locate all documents that mention the RDF statement `lipoprotein→affects→inflammatory_cells`. A similar query uses the provenance metadata to identify all RDF triples extracted from a particular document.
2) *Multi-document summarization*: Rank RDF statements from multiple documents using the confidence associated with each statement (indicated by the text mining tool).
3) *Question answering service*: Specify that the answers should be sourced only from reputable entities (for example, curated databases) or extracted from a journal article published recently (e.g., during the past year).
4) *Knowledge discovery service*: Using reasoning rules, implicit knowledge can be inferred from existing RDF triples in the BKR project. Often, provenance of the original triples is required to accurately interpret new triples. The application of reasoning rules can also be restricted to a specific set of RDF triples based on their provenance.

To address the above requirements, BKR collects the provenance information associated with an RDF triple at two levels. At the first level, provenance information directly associated with a RDF triple is collected, including the source of the triple (journal article, database) or some confidence value associated with it. At the second level, BKR records additional provenance information collectively associated with a set of triples. For example, all triples extracted from a given journal article inherit the date of publication, author names, and set of index terms of this particular journal article (for example, in Medline [124]).

The RDF reification vocabulary [24] has been traditionally used by Semantic Web applications to track provenance in RDF documents. The RDF reification vocabulary consists of the four terms `rdf:Statement`,4 `rdf:subject`, `rdf:predicate`, and `rdf:object`. Figure 6-1 illustrates the two levels of provenance recorded for the triple "`lipoprotein→affects→inflammatory_cells`" using RDF reification. A variety of problems have been identified in the use of RDF reification vocabulary for provenance tracking in Semantic Web applications.

Limitations of the RDF Reification and Related Approaches

The limitations of the RDF reification vocabulary are discussed along two dimensions, namely, (a) formal semantics, and (b) implementation issues for real world applications. The RDF specification [27] states that the RDF formal semantics does not extend to the reification vocabulary, and the intended interpretation of an RDF document using reification is application dependent (i.e., it may vary across applications) [24]. In addition to limited formal semantics, the RDF syntax does not provide a mechanism

[4] The rdf namespace represents the http://www.w3.org/1999/02/22-rdf-syntax-ns Internationalized Resource Identifier (IRI).

to link the reified triple to the RDF statement itself [24]. For example, there is no support in the RDF syntax to link "TripleID1432" in Figure 6-1 to the triple `lipoprotein→affects→inflammatory_cells`. The lack of support for consistent interpretation of RDF documents using reification is a significant challenge for scientific projects such as BKR that aim to serve a large community of researchers and to support multiple applications. The RDF specification describes a "conventional use" of the reification vocabulary [24] where "the subject of a reification triple" is a specific RDF triple in a particular RDF document and not any RDF triple (that may also have the same subject (S), predicate (P), and object (O)). Specifically, the assertion `TripleID1432→derives_from→PMID17209178` in Figure 6-1 is not applicable to all triples sharing the same S, P, O.

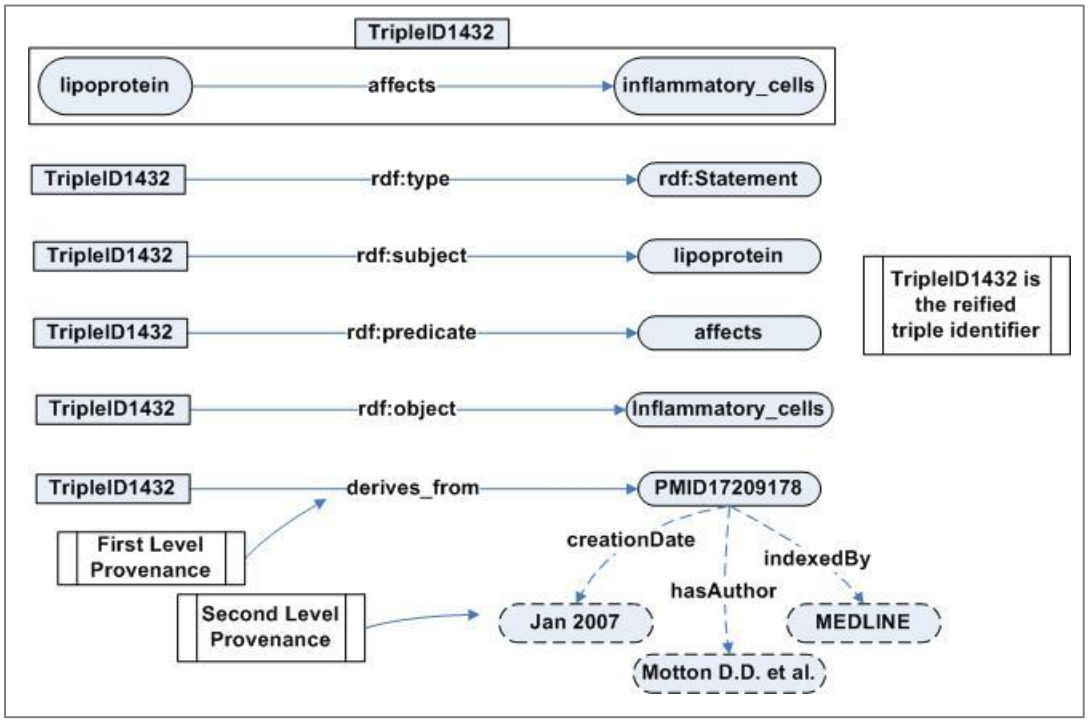


Figure 6-1 Provenance tracking on the Semantic Web using RDF reification

Inference rules are an important component of Semantic Web applications, especially for knowledge discovery tasks in projects such as the BKR. But, the RDF specification states that entailment rules do not hold between an RDF triple and its reification [27]. Further, the use of blank nodes, which have no "global meaning" outside a particular RDF graph [27] and have no corresponding real world entities in scientific domains, is a significant challenge to Semantic Web applications relying on reification. The use of blank nodes makes it difficult to use reasoning [125] and increases the complexity of query patterns

since the queries have to explicitly take into account an extra entity (that cannot be "typed" as instance of domain ontology class) in the query pattern.

We now describe the implementation specific limitation of RDF reification. Though incorporating additional metadata descriptions in form of provenance information necessarily increases the total size of an RDF document, the RDF reification approach leads to a disproportionate increase in the total size of the RDF document without corresponding enhancement in information content of the RDF document. For example, as illustrated in Figure 6-1, reification of a single RDF triple leads to the creation of four extra RDF triples that do not model any provenance-related information but are merely artifacts of the RDF syntax. This would adversely affect the scalability of large projects, such as BKR, that track provenance of hundreds of millions of RDF triples.

We now briefly describe two approaches, namely RDF named graph [126] and RDF molecule [127], that enable RDF provenance tracking at different levels of granularity. The named graph approach, part of the RDF specification, associates an identifier to an RDF graph that allows applications to make assertions about a set of RDF triples contained in the graph [126]. The named graph approach also defined the syntax, semantics and its relationship to RDF triples. The limitations of the named graph include its coarse granularity (that makes it impractical for use in real world applications) and the use of blank nodes. The RDF molecule is a similar approach to track provenance information at a finer level of granularity through lossless decomposition of a RDF graph to identify sub-graphs but not for a triple, using blank nodes [127]. In [128], a generalization of the RDF named graphs is proposed called "colored RDF triples" that uses a semi-group structure to reason over provenance information, but this work does not address the primary disadvantage of the named graph approach, that is, use of blank nodes.

In this chapter, we introduce a new approach called Provenance Context Entity (PaCE) to enable provenance tracking in Semantic Web applications using neither RDF reification vocabulary nor blank nodes. The PaCE approach creates the S, P, O RDF entities that reflect the provenance requirements of a Semantic Web application. PaCE is part of a broader framework for provenance management in scientific applications that was introduced in previous chapters (Chapter 2, 3, and 4).

6.2 Foundations of Provenance Context Entity

The intuition for the PaCE approach is that the provenance associated with RDF statements provides the necessary contextual information for applications to interpret two RDF statements to be equivalent or distinct. Contexts as formal objects have long been used in Artificial Intelligence (AI) applications, such as Cyc [129] and also to a limited extent in the Semantic Web, to facilitate processing of information that do not have a global frame of reference [130]. The next section reviews the existing work on context theory in AI.

6.2.1 Context Theory in AI

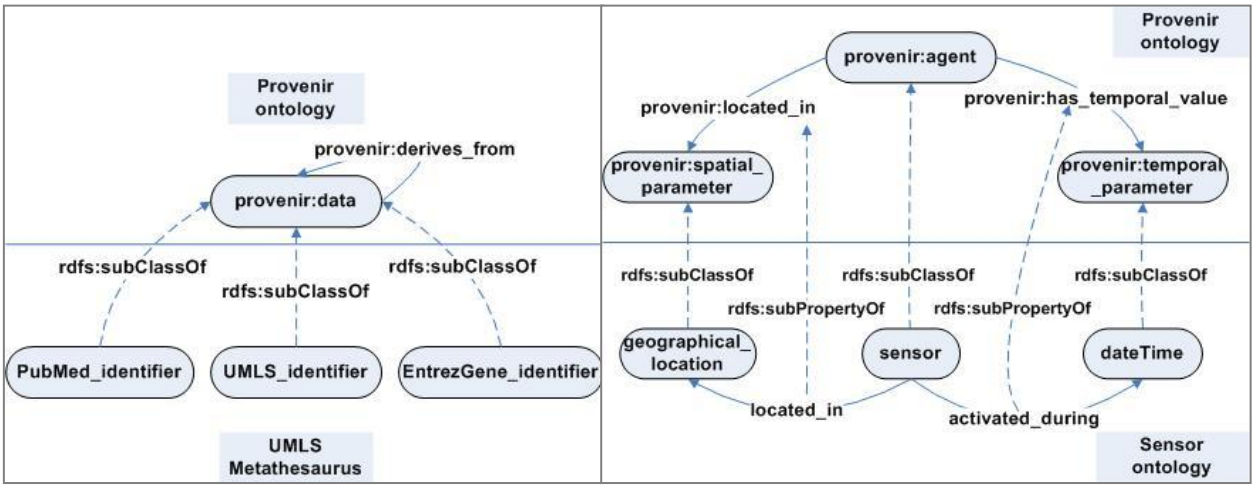
Contexts were introduced as formal objects in AI systems in the 1990s [131] [129] to allow applications to process statements only in specific frames of reference. Using the construct *ist* (*c*, *p*), which asserts that a statement *p* is true in a context *c*, context theory also defined mechanisms called "lifting rules" to process statements in different contexts [131] [129]. Various advantages of using contexts include (a) ability to make domain specific assumptions, (b) selection of a manageable subset of the knowledge base, and (c) maintaining consistency within a context without the need for maintaining global consistency [129]. There has been a lot of work in context research including appropriate extensions to the model theory and description of the associated computational complexity [132-134].

Contexts have been introduced for use in the Semantic Web to address challenges faced by data aggregation applications such as the TAP project [135]. For example, two apparently contradictory statements, "John Kennedy is president of USA" and "Barack Obama is president of USA", can be reconciled using contextual metadata describing the temporal information associated with the statements. The context mechanism in the TAP project associates a context with each Web data source and all information extracted from the source is assumed to be true (in the given context) [135]. The context for Semantic Web uses the *ist* (*c*, *p*) notation with appropriate extensions to the RDF model theory [135]. As discussed earlier in [126], these extensions to the RDF model theory require significant changes to existing implementations of Semantic Web inference systems. In contrast, existing implementations can process RDF documents that use the PaCE approach, which is defined in the next section, to track provenance information.

6.2.2 Provenance Context and RDF Generation

The contextual information in the BKR project consists of the provenance information about the source of an RDF statement, that is, the journal identifier or the UMLS identifier or the Entrez Gene identifier. In other words, this *provenance context* is a formal object instantiated in form of set of concepts and relationships that capture the necessary contextual provenance information to enable application to correctly interpret RDF statements. Similar to the provenance context defined in the BKR project, other Semantic Web applications can also define a relevant provenance context for interpreting their RDF dataset. For example, an application in the sensor domain can define its provenance context to consist of sensor used to collect data readings, the geographical location of the sensor, and the timestamp value associated with a data reading. To formalize the notion of provenance context we define it in terms of the Provenir ontology. An application can define its provenance context either in terms of the provenir ontology or in terms of a domain-specific provenance ontology, which extends provenir ontology. For example, the BKR project uses the UMLS Semantic Network (SN) [121] as the domain ontology and

hence defines its provenance context in terms of the SN (in section 4 we describe the mapping of relevant SN terms to the provenir ontology).



(a) (b)

Figure 6-2 Schematic representation of provenance context for the BKR project and a sensor application

Figure 6-2 (a) illustrates the BKR provenance context for the BKR project and Figure 6-2 (b) illustrates the provenance context for the sensor domain (consisting of the sensor identifier, the geographical location of a sensor, and date-time value associated with a sensor reading). It is important to note that the classes and properties used to define the provenance contexts for the BKR and sensor domain application can be linked to the provenir ontology using either "subclass" or "sub-property" relations. The use of the provenir ontology to define a provenance context has several advantages including the flexibility to model domain-specific provenance at a fine level of granularity, while ensuring consistent modeling and the support for RDF and OWL inferencing [27]. Further, applications can leverage a set of dedicated provenance query operators, defined in terms of the Provenir ontology. Though provenance context as a notion is derived from the context theory used in AI systems, it is distinct in terms of both formal semantics and implementation. These differences are listed below:

1. A provenance context is used only for generating the S, P, O of an RDF triple and this approach of generating "provenance-aware" RDF triples is called the Provenance Context Entity (PaCE) approach. In contrast, traditional AI systems use context primarily during processing or interpreting data.
2. The PaCE approach involves *apriori* use of the context object during RDF triple generation; hence it does not use the *ist* (*c*, *p*) construct to interpret RDF statements. In addition, unlike the context mechanism introduced in [135], the PaCE approach does not require extensive modifications to the RDF model theory.

3. The PaCE approach defines a single application-wide provenance context and unlike traditional AI systems does not include multiple context objects. Hence, the PaCE approach does not require use of lifting rules to process RDF statements in different contexts [129].

The PaCE approach allows an application to decide the level of granularity in modeling provenance of an RDF triple. For example, Figures 6-3 and 6-4 illustrates the three possible implementations of the PaCE approach in the BKR project that create distinct RDF triples extracted from two separate journal articles (though they share the same S, P, and O). The first implementation (Figure 6-3 (a)) is an exhaustive approach and explicitly links the S, P, and O to the source journal article and the second implementation (Figure 6-3 (b)) is a minimalist approach that links only the S of a RDF triple to the source article. Though the first implementation creates three provenance-specific triples, in contrast to just one triple by second implementation, there is no ambiguity in correctly interpreting the provenance of the triples. The second implementation, on the other hand, requires the application to make additional assumption, while processing the RDF triples, that the whole triple is extracted from the same source as the source of S. Hence, the additional complexity associated with the second implementation may make it unsuitable for some applications.

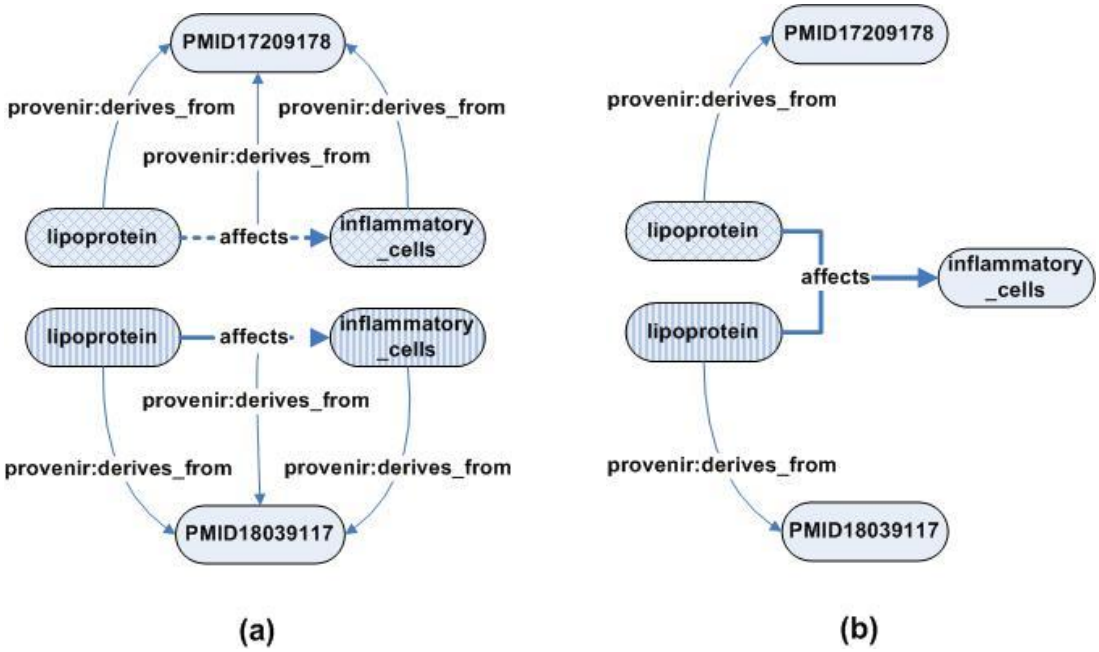


Figure 6-3 Implementation of the PaCE mechanism using (a) exhaustive approach, and (b) minimalist approach

The third implementation (Figure 6-4(c)) takes an intermediate approach that creates two additional provenance-specific triples but requires the application to assume that the source of the O is the same as the S, and P. Similar to the minimalist approach, this approach reduces the total number of provenance-specific RDF triples, but introduces additional complexity that may make this approach unsuitable for

some applications. The choice to associate explicit "`derives_from`" property with one particular RDF component (S or P or O) in the minimalist (Figure 6-3 (b)) and the intermediate (Figure 6-4(c)) is arbitrary and has minimal impact on the provenance tracking functionality of the application.

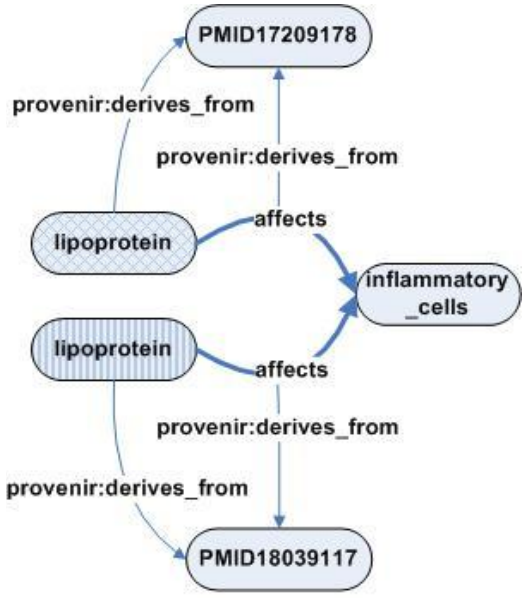


Figure 6-4 Implementation of the PaCE mechanism using the intermediate approach

PaCE approach requires the use of RDF reification vocabulary or the use of blank nodes. Further, the reification approach creates a total of six RDF triples (Figure 6-1) for each RDF triple, while the exhaustive implementation of the PaCE approach creates a total of four triples for one RDF triple. This difference in the total number of RDF triples generated by the reification approach versus the PaCE approach has significant impact on the scalability of applications incorporating provenance information in real world applications, such as the BKR project that includes millions of RDF triples. This significant difference in total number of provenance-related RDF triples generated using the PaCE approach as compared to the RDF reification approach is further discussed in Section 6.4 (Evaluation).

Overall, the PaCE approach is an incremental and simple mechanism that does not define additional vocabulary or require changes to existing RDF data stores. In addition, the PaCE approach does not require modifications to existing RDF or OWL inference systems used in many Semantic Web applications. We now introduce the formal specification of provenance context.

6.3 Model Theoretic Semantics of PaCE Inferencing

The primary motivating factor for defining the formal semantics of PaCE is to provide a way to determine the validity of the inferencing process for Semantic Web applications that use the PaCE approach to track

provenance. For example, the BKR project can derive a ranking of RDF statements extracted from journal articles by inferring the confidence value of an RDF statement from the precision and recall values indicated by a text mining tool. To define the formal semantics of PaCE we use model theory. Specifically, we build on the approach used to define the formal semantics for RDF and RDF Schema (RDFS) [27] to define the model theoretic semantics of PaCE. So our definition is based on the notions of *interpretations* and *models*, which are structures that enable us to capture information about truth values (true or false) of assertions [27]. In other words, if a particular interpretation I satisfies a specific assertion $s \in V$ then we call I a model of s and write $I \models s$ in this case (where V is a vocabulary and $\models$ is the so-called entailment relation).

Following [27], a *simple* interpretation I of a vocabulary V consists of

- a non empty set of *resources* IR that constitutes the domain or universe of I,
- a set of *properties* of I called IP,
- a function I_{EXT}: IP $\rightarrow 2^{IR \times IR}$ that maps each property in IP to a pair of resources in IR,
- a function I_S: V $\rightarrow$ IR $\cup$ IP which maps IRIs in V to the union of IR and IP,
- a function IL which maps typed literals from V to resources in IR, and
- a subset of IR called *set of literal values*, LV, containing all untyped literals from V,

Each interpretation I then gives rise to an interpretation function $\cdot^I$, which maps each triple (over IR, IP, V and LV) to a truth value (true or false) in a canonical way (see [27]) An interpretation I of a graph R is said to be a *model* of R if I maps each triple in R to the truth value true. We write $I \models R$ in this case. Simple interpretations allows us to define an entailment relation between graphs, that is, a graph R_1 (simply) entails a graph R_2 if every simple interpretation that is a model of R_1 is also a model of R_2. A simple interpretation of a vocabulary $V \cup V_{RDF} \cup V_{RDFS}$ is called an *RDFS interpretation* of V if it satisfies a number of additional constraints specified in [27]. We say that a graph R_1 RDFS-entails a graph R_2 if every RDFS interpretation that is a model of R_1 is also a model of R_2.

The definition of the model-theoretic semantics of PaCE is a straightforward modification of the existing RDFS semantics and allows us to infer additional provenance information for triples by virtue of having similar source. Let provenance context pc of an RDF triple α (= (S, P, O)) be the common object of the predicate `provenir:derives_from` associated with the triple. We define an *RDFS-PaCE-interpretation* I of a vocabulary V to be an RDFS-interpretation of the vocabulary $V \cup V_{PaCE}$ that satisfies the following additional condition (meta-rule):

- For RDF triples $\alpha = (S_1,P_1,O_1)$ and $\beta = (S_2,P_2,O_2)$, (provenance-determined) predicates *p* and entities *v*,

  if $pc(\alpha) = pc(\beta)$

  then $(S_1, p, v) = (S_2, p, v)$ and, $(P_1, p, v) = (P_2, p, v)$ and, $(O_1, p, v) = (O_2, p, v)$

Provenance-determined predicates and entities are specific to the application domain.

Furthermore, a graph R_1 PaCE-entails a graph R_2 if every *RDFS-PaCE-interpretation* that is a model of R_1 is also a model of R_2. To illustrate the PaCE inference process, we consider two RDF statements in the BKR project (Figure 6-5). Given that the two RDF statements have equal provenance contexts (PubMed identifier: PMID17209178) additional provenance information, such as the confidence score (formalized via provenance-related predicate `has_confidence_value` and value `confidence_value_2`), associated with one of the triples can be inferred for the other RDF triple (`flow_cytometry→measures→interleukin-1_beta`) denoted by dotted arrows in Figure 6-5.

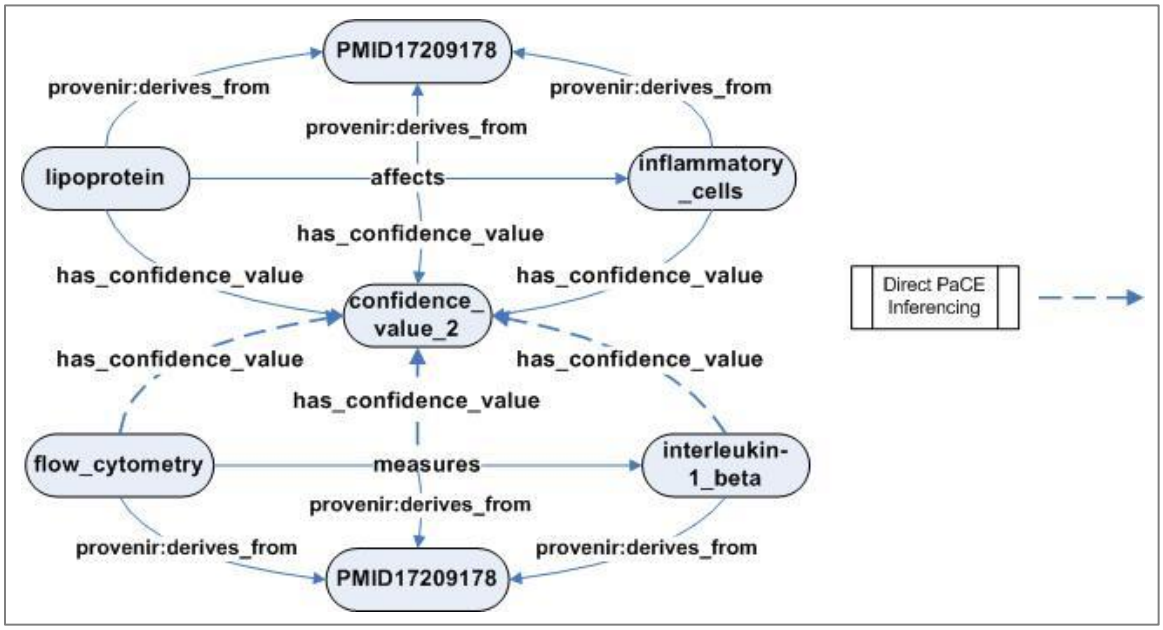


Figure 6-5 PaCE inferencing

We note that PaCE-entailment is strictly stronger than RDFS-entailment in the sense that all inferences which can be drawn using simple, RDF, or RDFS-entailment are also PaCE entailments. This is a deliberately conservative step on top of the existing Semantic Web recommendations that enables PaCE to be compatible with existing OWL and RDF tools and applications, and also allows implementing the PaCE-semantics by making reference to RDF reasoners as black boxes. In the next section, we describe the implementation of the BKR project using the PaCE approach.

6.4 Implementation: Using PaCE Approach in the BKR Project

Implementing the PaCE approach in BKR project involves two steps, namely, (a) Extending the provenir ontology with domain-specific concepts that serve as a reference for the definition of contextualized

instances in the BKR, and (b) generating instance-level RDF triples leveraging the provenance model for data from several sources. We first describe the extension of the provenir ontology.

6.4.1 Extending the Provenir Ontology with Domain-specific Concepts

The provenir ontology provides a domain-independent model for provenance, which needs to be extended with domain-specific concepts in order to support the creation of contextualized instances. We use the Unified Medical Language System (UMLS) as our main source of biomedical concepts [33]. More specifically, high-level categories (semantic types) from the UMLS Semantic Network can be integrated as subclasses (`rdfs:subClassOf`) of the provenir classes `provenir:data` or `provenir:event`. In turn, the two million concepts from the larger UMLS Metathesaurus are integrated as subclasses of the semantic types, using the categorization link provided by the UMLS. The provenir ontology is also extended with some 27,000 genes from Entrez Gene for better coverage of genomic entities.

Instances in the BKR are defined in reference to these domain-specific concepts. Analogously, instance-level predicates in the BKR are defined as sub-properties (`rdfs:subPropertyOf`) of the 53 named relationships provided by the UMLS Semantic Network. A mapping between instance-level and Semantic Network predicates was created manually. In addition to links (`rdf:type`) between instances from the BKR and the corresponding classes, the definition of BKR provenance context in enabled through the `provenir:derives_from` relationship linking a triple to its source. The `provenir:derives_from` property is adapted from the `derives_from` property defined in the upper-level Relation Ontology and is used to model the "derivation history of data entities as a chain or pathway" [83].

6.4.2 Generating RDF Triples using the PaCE Approach

Contextualized RDF triples in the BKR represent knowledge extracted from the biomedical literature, as well as relations from the UMLS Metathesaurus. RDF triple entities (S, P, O) are identified using an unique identifier called the Uniform Resources Identifier (URI). A practical challenge for implementing the PaCE approach in the BKR is to formulate an appropriate provenance context-based URI (URI_p) scheme that also conforms to best practices of creating URIs for the Semantic Web, including support for use of HTTP protocol [136].

The design principle of URI_p is to incorporate a "provenance context string" as the identifying reference of an entity and is a variation of the "reference by description" approach that uses a set of description to identify an entity [136]. The syntax for URI_p consists of the <*base URI*>, the <*provenance context string*>, and the <*entity name*>. For example, the URI_p for the entity lipoprotein is

http://mor.nlm.nih.gov/bkr/PUBMED_17209178/lipoprotein where the PUBMED_17209178 provenance context string identifies the source of a specific instance of lipoprotein.

This approach to create URIs for RDF entities also enables BKR (and other Semantic Web applications using the PaCE approach) to group together entities with the same provenance context. For example,

- `http://mor.nlm.nih.gov/bkr/PUBMED_17209178/lipoprotein`
- `http://mor.nlm.nih.gov/bkr/PUBMED_17209178/affects`
- `http://mor.nlm.nih.gov/bkr/PUBMED_17209178/ inflammatory_cells`

are entities extracted from the same journal article. Using this URI scheme, RDF statements were generated for the original triples (extracted from the biomedical literature by a text-mining application or found in the UMLS Metathesaurus).

In the next section, we evaluate the implementation of the PaCE approach to track provenance in the BKR project as compared to the RDF reification approach.

6.5 Evaluation

The objective of our experiment is to evaluate the advantages of using the PaCE approach in place of the RDF reification approach to track provenance in the BKR project. Three specific aspects are investigated:

1. Measure the burden of representing provenance information, in number of triples required, compared to a "base dataset" (B) with no provenance information
2. Analyze the performance of four BKR provenance queries
3. Demonstrate the use of provenance information to support analytical queries in the BKR project and measure the associated cost in performance

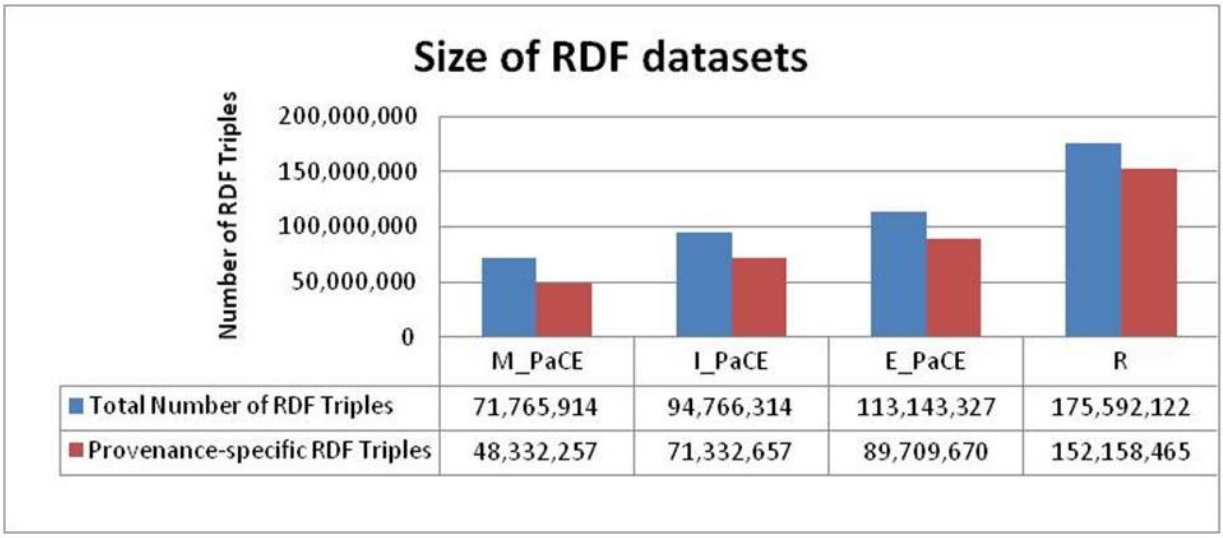
The base dataset (B) comprises of 23,433,657 RDF triples extracted from two sources: the biomedical literature (PubMed) and the UMLS Metathesaurus.

The open source Virtuoso RDF store version 06.00.3123 was used for the experiments running on a Dell 2950 server (Dual Xeon processor) with 8GB of memory. A total of 500,000 9kB buffers were allocated to Virtuoso RDF store.

6.5.1 Number of Provenance-specific RDF Triples Generated

To evaluate the number of provenance-specific RDF triples generated using the two approaches, we augment the base dataset B with provenance information representing the source information of each triple. For the PaCE approach, we create three datasets representing the exhaustive (E_PaCE), minimalist (M_PaCE), and intermediate (I_PaCE) approaches illustrated in Figure 6-3 (a), 6-3 (b) and 6-4 (c), respectively. For the RDF reification dataset (R), we use the standard method (presented in Section 1).

Figure 6-6 shows that the reification approach requires twice as many RDF triples (~152 million) for the representation of provenance information compared to the E_PaCE approach (~89 million). This 49% difference between E_PaCE and R represents a significant reduction in storage requirements (~85 million fewer triples) for the BKR project and, more generally, clearly demonstrates the practical benefits of using the PaCE approach over reification to track provenance in Semantic Web applications. Analogously, the M_PaCE and I_PaCE approaches create 72% and 59% fewer provenance-specific triples compared to the reification approach.



	M_PaCE	I_PaCE	E_PaCE	R
Total Number of RDF Triples	71,765,914	94,766,314	113,143,327	175,592,122
Provenance-specific RDF Triples	48,332,257	71,332,657	89,709,670	152,158,465

Figure 6-6 The relative number of provenance-specific triples created using PaCE and RDF reification

6.5.2 Performance of Provenance Queries

We use four representative categories of provenance queries in the BKR project to evaluate the query performance for the four datasets (E_PaCE, M_PaCE, I_PaCE and Reification). We describe the *pattern* of the four queries and their significance in the BKR project:

Query Pattern 1: List all the RDF triples extracted from a given journal article (e.g., journal article identified by PMID17209178). This query is used to retrieve all the triples from a given source.

Query Pattern 2: List all the journal articles from which a given RDF triple was extracted (e.g., `lipoprotein→affects→inflammatory cells`). This query identifies the source(s) of a given triple.

Query Pattern 3: Count the number of triples in each source (biomedical literature and UMLS Metathesaurus) for the therapeutic use (predicate = treats) of a given drug (e.g., Thalidomide). This complex query illustrates the use of the BKR as a knowledge base for a query answering application (e.g., which diseases are treated by a particular drug?).

Query Pattern 4: Count the number of journal articles published between two dates (e.f., 2000-01-01 and 2000-12-31) for a given triple (e.g., thalidomide → treats → multiple myeloma). This typical information retrieval query leverages the provenance information associated with each triple. A more complex version of this query is used Section 5.2 for time series analysis.

We conducted the query performance evaluation in two phases. In the first phase the four queries are evaluated for fixed values, namely the value underlined in the query description above. In the second phase, queries are evaluated using a larger set of values. The queries are expressed in SPARQL syntax, the RDF query language [69], and primarily utilize the SPARQL basic graph patterns (BGP) with FILTER conditions. The queries are not listed in the paper due to space constraints and are available online along with the result set.5 The numbers reported for the "fixed" value queries (first phase) are the average of last 5 of a total of 20 runs. The first phase of the evaluation starts with a "cold" cache for each query pattern.

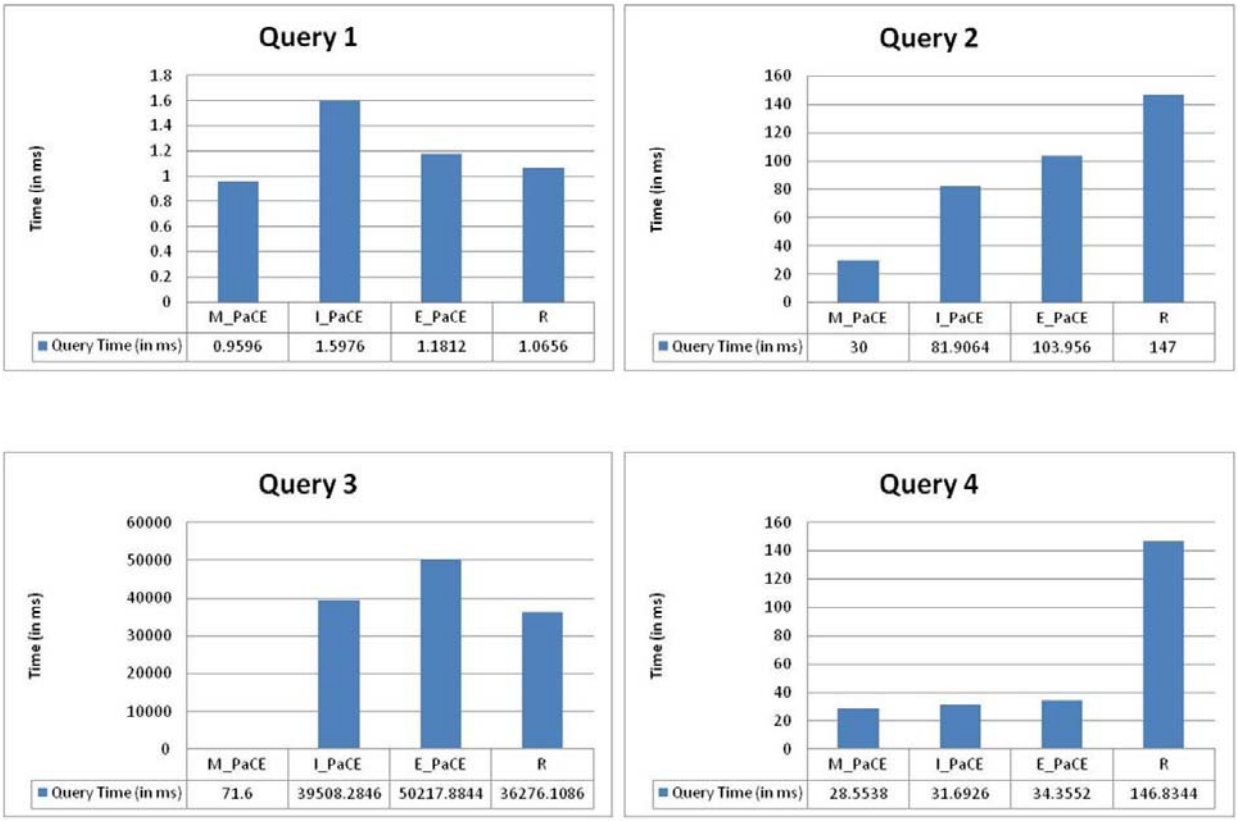


Figure 6-7 Query performance for fixed values

The results in Figure 6-7 demonstrate that query performance for PaCE is generally better than or similar to reification. As expected (Figure 6-7), M_PaCE generally performs better than I_PaCE, and

[5] Query and result set available at: http://wiki.knoesis.org/index.php/ProvenanceContextEntity

I_PaCE better than E_PaCE. However, reification performs better than I_PaCE for *Query 1* and better than both I_PaCE and E_PaCE for *Query 3*. *Query 4* is a complex query that uses the SPARQL FILTER to restrict publication dates to a particular range (January 1 to December 31, 2000). In this query, the query performance for E_PaCE is more than two orders of magnitude better than for R.

In the second phase of the evaluation, we aim to reflect the real-world requirements of the BKR project. Toward this end, each of the four query patterns is executed with different values, as if by different users. In practice, we use sets of 100 values for each query pattern. The resulting set of 100 queries is run 5 times (immediately following the first phase of evaluation for each dataset) and the average of the 100 queries for the last run is presented (Figure 6-8).

The results confirm the trend seen in the first phase of evaluation, with the added observation that for *Query Pattern 3* the difference between E_PaCE and R has decreased (R no longer outperforms E_PaCE significantly). In contrast, for the complex *Query Pattern 4*, the query performance for E_PaCE has further improved and is more than three orders of magnitude better than for R. The second phase of evaluation also confirms that in a real-world scenario the query performance of PaCE is comparable to reification for simple provenance queries.

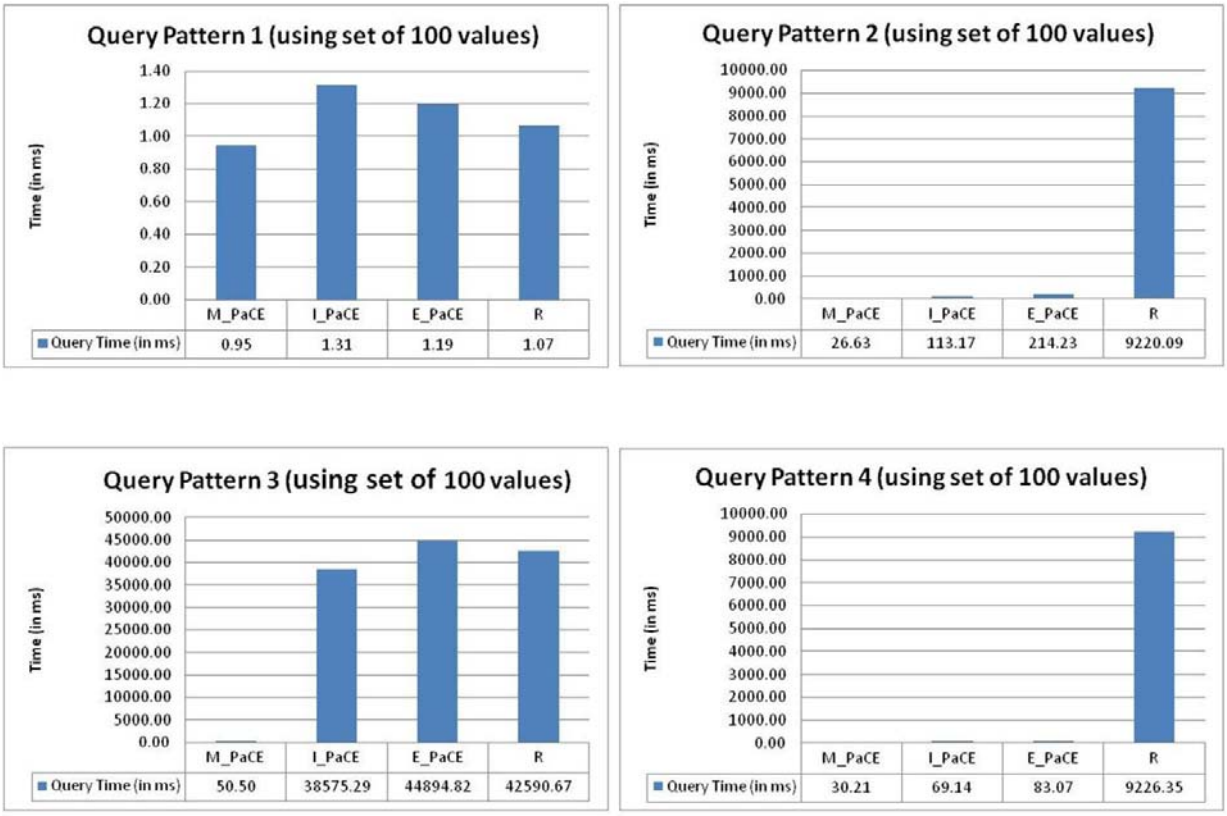
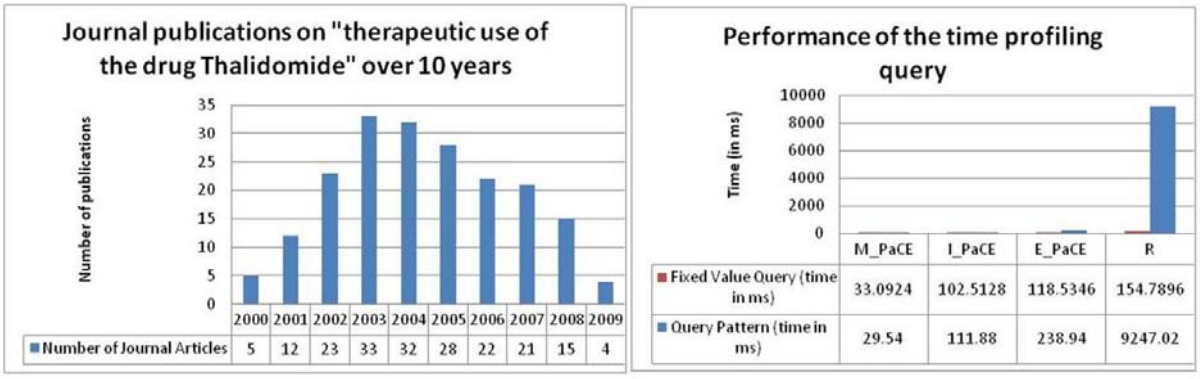


Figure 6-8 Query performance for query patterns using a set of 100 values

In the next section, we evaluate the query performance for an analytical query in the BKR project that uses provenance information for identify the publication pattern of journal articles on a specific topic of interest.

6.5.3 Performance of Provenance Queries

An important objective for many applications and funding agencies is to understand the trend in research focused on a specific topic in biomedicine over a period of time. We extend the *Query Pattern 4* discussed in the previous section to define a query that collates the number of journal articles published over a period of 10 years (i.e., the span of the current BKR). As an example, we focus on mentions of the therapeutic use of the drug Thalidomide over time. This query translates to a complex SPARQL query that uses functions to aggregate number of publications per year. Figure 6-9 (a) shows a histogram created directly from the query results. The query performance is similar to what was observed for *Query Pattern 4*, that is, E_PaCE is three orders of magnitude faster than R (Figure 6-9(b)). This example demonstrates the feasibility of using RDF and SPARQL for representing and exploiting provenance information in large triple stores serving real-world applications.



(a) **(b)**

Figure 6-9 (a) Result of time profiling provenance query for the therapeutic use of drug Thalidomide, (b) performance of the query using RDF reification and PaCE mechanism

We show that that challenge of provenance tracking in RDF datasets can be effectively and efficiently addressed by using the PaCE approach in place of the RDF reification vocabulary. The PaCE approach addresses many of the issues associated with RDF reification, including lack of formal semantics, use of blank nodes, and application-dependent interpretation of RDF documents. The PaCE approach uses the formal objects called provenance contexts that are defined in terms of the provenir upper-level provenance ontology to create provenance-aware RDF triple entities of S, P, and O. The model-theoretic semantics of PaCE is defined through a simple extension of the existing RDFS formal semantics. We implemented the PaCE approach in the BKR project. The evaluations demonstrate that using the PaCE

approach to create provenance-specific RDF triples not only reduces the number of triples by at least 49% but also improves the performance of complex provenance queries by three orders of magnitude. We plan to extend BKR with data from additional sources and use the PaCE approach for provenance tracking.

6.6 Conclusions

We show that that challenge of provenance tracking in RDF datasets can be effectively and efficiently addressed by using the PaCE approach in place of the RDF reification vocabulary. The PaCE approach addresses many of the issues associated with RDF reification, including lack of formal semantics, use of blank nodes, and application-dependent interpretation of RDF documents. The PaCE approach uses the formal objects called provenance contexts that are defined in terms of the provenir upper-level provenance ontology to create provenance-aware RDF triple entities of S, P, and O. The model-theoretic semantics of PaCE is defined through a simple extension of the existing RDFS formal semantics. We implemented the PaCE approach in the BKR project. The evaluations demonstrate that using the PaCE approach to create provenance-specific RDF triples not only reduces the number of triples by at least 49% but also improves the performance of complex provenance queries by three orders of magnitude.

Chapter 7

Application: A Framework for Provenance Management in Translational Research

Introduction

The NIH *Translational Research* roadmap seeks to strengthen the research infrastructure for accelerating the delivery of "bench-side" discoveries to patient's "bedside". The key notion of translational research is the flow of information resources (experiment data, publications/literature, clinical trial data, or patient records) across organizations, domains, and projects that impacts both patient care and (through a feedback process) basic research. This necessitates keeping track of provenance metadata describing the complete life cycle of information resources from the point of their creation to intermediate processing, and finally their end use. For example, provenance information describing the experiment design includes details about the biological or technical replication (RNA extracts or cDNA clones) in microarray experiments [137], the type of parasite used to create an avirulent (non-virulent) strain in a gene knockout experiment [105], or demographic information of subjects used in a clinical trial [138] are essential for drawing valid conclusions from the relevant information resources. Similarly, provenance information about the experiment platform (type of instruments used, instrument settings) and the tools used to process or analyze data (algorithms, statistical software), is necessary for ensuring data quality and thereby association of confidence or trust with the results that account for data lineage.

Traditionally, provenance has been collected manually or using custom-built software tools that have several drawbacks, including the difficulty in ensuring that adequate amount of provenance is collected and support for provenance interoperability across projects. Further, the use of high-throughput data generation technologies in basic research, such as high-throughput sequencing, microarrays, mass spectrometry (ms), and nuclear magnetic resonance (NMR), along with large datasets in clinical research, such as electronic health care records (EHR) and clinical trials data, are introducing additional challenges for the traditional approaches to provenance management, including scalability and support for knowledge discovery tasks using automated reasoning.

Broadly, we can classify the challenges of provenance management in translational research along four dimensions, namely

(a) **Collecting provenance** information in high throughput environments that is also adequate to support complex queries,

(b) **Representing provenance information** using a model that supports interoperability across projects, is expressive enough to capture the complexities of a specific domain (*domain semantics*), and allows use of reasoning for provenance analysis,

(c) A dedicated **query infrastructure** that allows composition of provenance queries with minimal user effort, addresses the requirements specific to provenance queries (support for transitive closure, neighborhood retrieval, computing reachability), and

(d) **Storing provenance** using highly scalable implementation to support complex user queries over large volumes of data.

In this chapter, we describe a provenance management framework that addresses these four aspects in an integrated manner and can be adapted for use in a wide variety of application domains in translational research. The framework and its implementation are discussed in the context of an exemplar biomedical research project on the human parasite *Trypanosoma Cruzi* (*T.cruzi*). We describe this project in the next section.

7.2 The Semantic Problem Solving Environment for *T.cruzi* project

T.cruzi is the principal causative agent of the human Chagas disease and affects approximately 18 million people, predominantly in Latin America. About 40 percent of these affected persons are predicted to eventually suffer from Chagas disease, which is the leading cause of heart disease and sudden death in middle-aged adults in the region. Research in *T.cruzi* has reached a critical juncture with the publication of its genome in 2005 [139] and can potentially improve human health significantly. But, mirroring the challenges in other translational research projects, current research efforts in *T.cruzi* to identify vaccine candidates in *T.cruzi* and development of diagnostic techniques for identification of best antigens, depend on analysis of vast amounts of information from diverse sources. The Semantic Problem Solving Environment (SPSE) for *T.cruzi* project is a collaborative effort, between the Ohio Center of Excellence for Knowledge-enabled Computing (Kno.e.sis) at the Wright State University, the Center for Tropical and Emerging Global Diseases at the University of Georgia, and the National Center for Biomedical Ontologies (NCBO) at the Stanford University, to address this challenge through creation of an integrated environment that allows dynamic ontology-driven integration of multi-modal local and public data to answer biological queries at multiple levels of granularity [105].[6]

In contrast to the traditional information integration approaches, the *T.cruzi* SPSE project is systematically incorporating detailed provenance of the information resources that are added to its parasite knowledge repository (PKR) resource. The provenance information is leveraged to achieve multiple objectives and these are discussed in the next section.

[6] The *T.cruzi* SPSE project is funded by the National Heart Lung and Blood Institute of the National Institutes of Health (NIH).

7.2.1 Provenance requirements in the *T.cruzi* SPSE project

The *T.cruzi* SPSE PKR resource integrates information resources represented in heterogeneous formats and from multiple sources, such as experiment data from reverse genetics protocols to create avirulent strains of *T.cruzi*, data stored in structured databases such as the whole-organism proteomics data for the four life cycle stages of *T.cruzi* [139], and plain text comments by researchers regarding an ongoing or newly started experiment run/project. The provenance information associated with each of these sources includes version information of a database, the name of the researcher and date on which the comments were created, and the conditions under which an experiment dataset was produced. Among these three examples, provenance information associated with the experiment datasets are the most complex to manage. An important approach to the study of *T.cruzi* infection is the use of reverse genetics to create avirulent (non-virulent) strains of the *T.cruzi* parasite in the laboratory. The creation of such parasite strains requires the identification of genes that control a core biochemical function. These genes can be deleted from the genome of the parasite (gene "knock-out") in order to ablate the biochemical function, possibly resulting in an avirulent strain. The two experiment processes used in creation of *T.cruzi* avirulent strains are (a) Gene Knockout (GKO) Protocol, and (b) Strain Project (SP) Protocol.

Given a list of genes for creation of potential avirulent *T.cruzi* strains, each gene forms an input to the GKO experiment protocol. To totally ablate (or at a minimum reduce) the function of genes, each of the alleles of the genes are targets of knock-out (Figure 7-1). The output of the GKO experiment protocol is a "knockout construct plasmid", which is created using the appropriate sequences of the target gene and a chosen antibiotic resistance gene. This plasmid is used in the SP experiment protocol to create a new strain of *T.cruzi* (Figure 7-1).

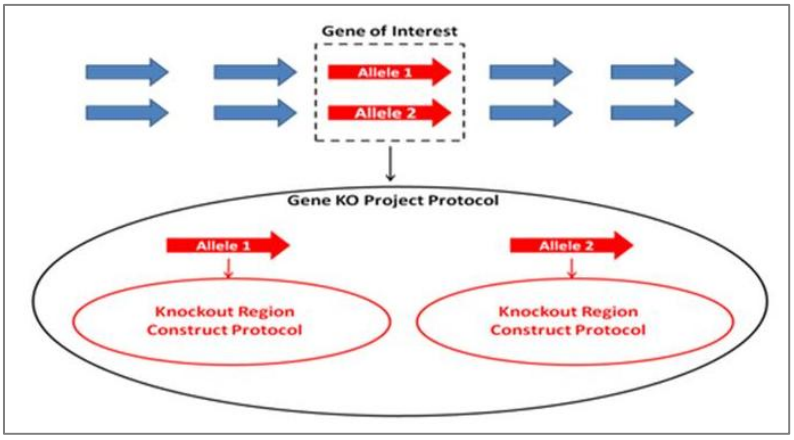


Figure 7-1 Alleles of target genes are used to create knockout plasmids

The SP Protocol is composed of three sub-processes (described in Figure 7-2) namely, Transfection, Drug Selection, and Cloning. Briefly, during transfection the Knockout Construct Plasmid will replace

the target gene in the *T. cruzi* genome with a selected antibiotic resistance gene resulting in a "Transfected Sample". The expression of the antibiotic resistance gene will allow parasites that were successfully transfected to survive drug treatment (Selection) with an antibiotic such as Neomycin. Researchers treat the Transfected Sample with the antibiotic for the period of time that kills all parasites in a non-transfected sample. Individual parasites within the resulting "Selected Sample" are then cloned to create "Cloned Samples" which are then used to infect model animals such as mice to assess strain phenotype and attenuation.

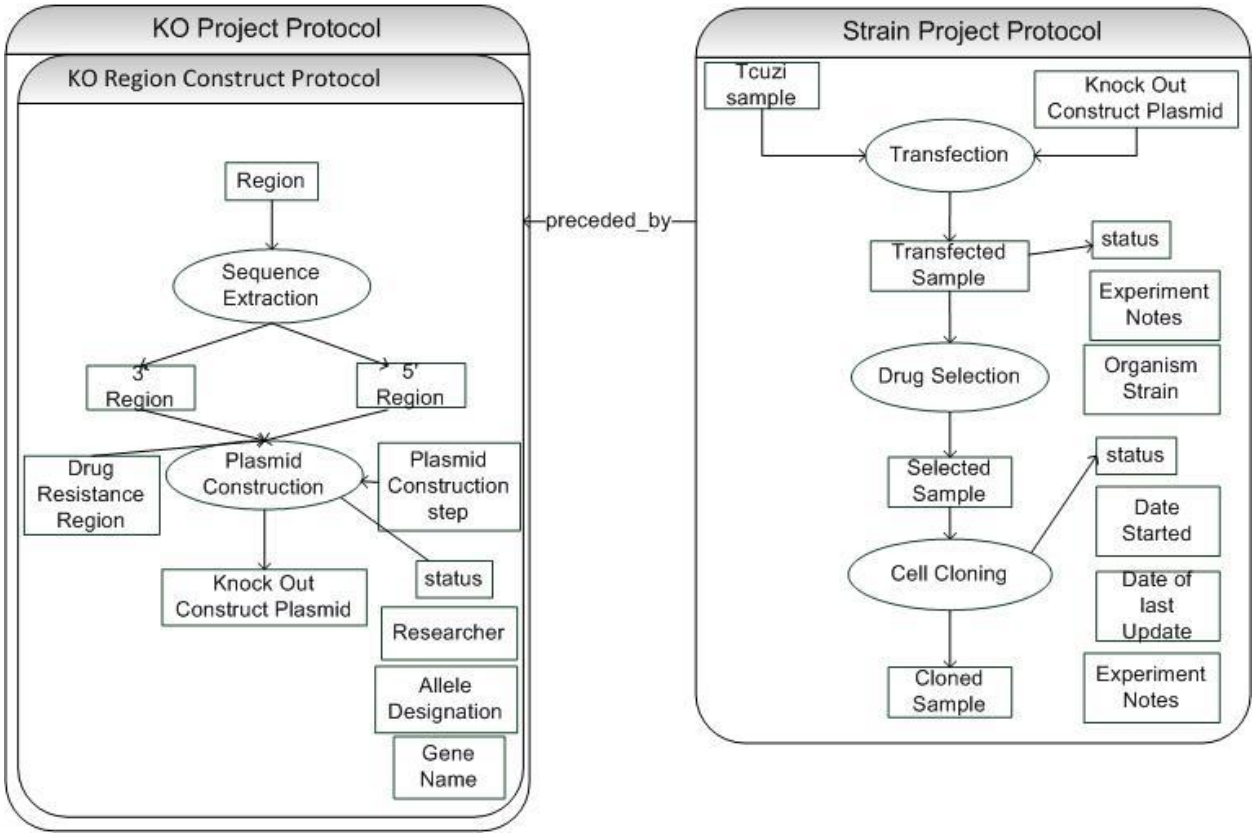


Figure 7-2 Schematic representation of GKO and SP experiment protocols

7.2.2 Querying Provenance Information of Experiment Data

The provenance information collected during GKO and SP experiments is used by multiple users with different requirements:

1) Technicians performing the lab-related work,
2) Project managers or principal investigators who want to track progress and/or view strains successfully created,
3) New researchers such as visiting faculty or post-docs who want to learn the lab-specific methods, and

4) Researchers in the parasite research community who can infer phenotype of the related organisms that they study from the work done on *T. cruzi*.

The current informatics infrastructure in the Tarleton research group is built using multiple relational databases that are accessed via custom web pages to store, track, and view data for a project. We describe how an example set of provenance queries are executed using the existing infrastructure.

Query 1: *List all groups using"target_region_plasmid_Tc00.1047053504033.170_1" target region plasmid*?

Current Approach: This query cannot be performed by a user using the current infrastructure in the Tarleton research group. The informatics specialist has to search for data from different databases and then create a set of customized SQL queries and scripts to answer the query.

Query 2: *Find the name of the researcher who created the knockout plasmid "plasmid66"?*

Current Approach: Answering this query requires access to three tables from two database schemas and use of PHP-based web tools. A custom query builder tool is used to search for plasmids with identifier "plasmid66". Next, in a three step process, including searching for "Strain" record associated with the given plasmid identifier and gene details, the name of the researcher is located.

Query 3: "c*loned_sample66" is not episomal. How many transfection attempts are associated with this sample?*

Current Approach: Using a custom query builder tool, a SQL query joining two tables is generated to list the strains with cloned samples with confirmed episomal insertion of the plasmid. From this list the user can obtain the number of transfection attempts to create the strain.

Query 4: *Which gene was used create the cloned sample "*c*loned_sample66"*?

Current Approach: To answer this query the researcher again uses a custom query builder to select the "KO Gene Name" in a tabular result view and search for "cloned_sample66". The custom query builder performs the joins necessary to combine data from three tables and creates the SQL query automatically. However, changes to the underlying database require modification of the PHP code by the informatics specialist.

These example queries demonstrate the limitations of current infrastructure that either cannot answer a query (Query 1) or requires the user to follow a multi-step process to retrieve the result. These limitations, especially the manual effort required, assume significance in a high-throughput experiment environment with multiple concurrent projects and the need to integrate provenance information across projects to guide future research. In the next section, we describe the ontology-driven provenance management infrastructure that has been created to address these challenges.

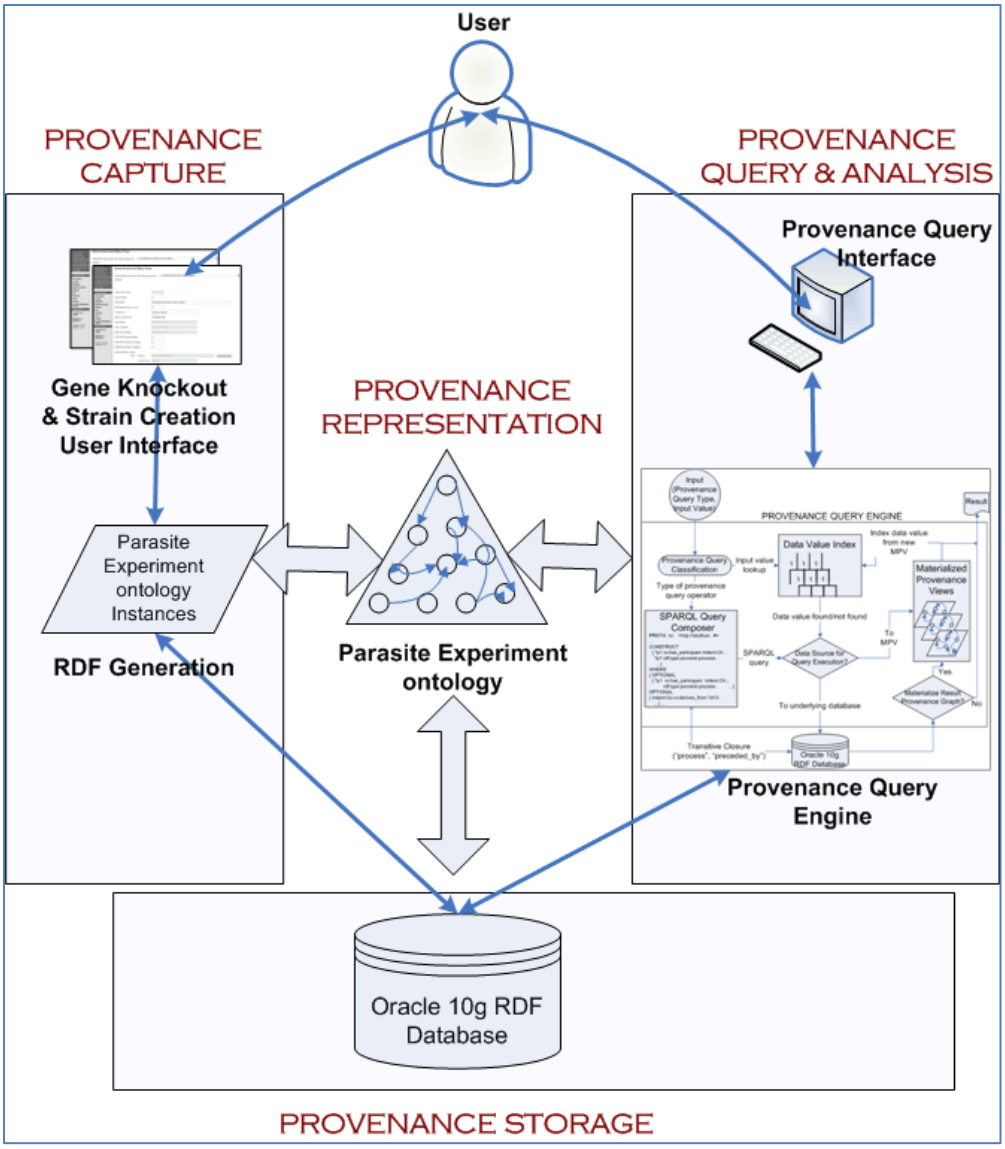


Figure 7-3 The architecture of the *T.cruzi* provenance system addressing four aspects of provenance management

7.3 *T.cruzi* Provenance Management System

The *T.cruzi* Provenance Management System (PMS) infrastructure addresses four aspects of provenance management in the *T.cruzi* SPSE project (Figure 7-3):

1. **Provenance Capture** – The provenance information associated with SP and GKO experiment protocols are captured via web pages used by researchers during an experiment. This data is transformed into RDF instance data corresponding to the PE ontology schema.
2. **Provenance Representation** – The parasite experiment (PE) ontology is used to model the provenance information of GKO and SP experiment protocols. The integrated provenance

information, from both these experiment protocols, is represented as "ground" RDF graph, that is, without any blank nodes [27].

3. **Provenance Storage** – The provenance information is stored in an Oracle 10g (release 10.2.0.3.0) RDF database management system (DBMS). Oracle 10g was chosen due to its widespread use in biomedical informatics applications [140] [141] and it satisfied the requirements of the *T.cruzi* PMS. For example, it supports the full SPARQL query specification [69], use of RDFS [27] as well as user-defined reasoning rules, and is a proven platform for large scale RDF storage [114].We note that the *T.cruzi* PMS can be implemented over any RDF DBMS that support the above listed requirements.
4. **Provenance Query Analysis** – In addition to storage of provenance information, the *T.cruzi* PMS supports querying of provenance information, using a set of specialized provenance query operators. The query operators are implemented in a query engine deployed over the Oracle RDF database.

7.3.1 The Parasite Experiment Ontology

In addition to the description in Chapter 3 about the creation of the Parasite Experiment ontology (PEO), by extending the Provenir ontology, we also briefly describe the interoperability of PEO with existing biomedical ontologies. The NCBO currently lists 137 publicly available biomedical ontologies [11] and it is important that new ontologies, such as PE ontology, are interoperable with these existing ontologies. Hence, the PE ontology re-uses relevant classes and relationships from four public ontologies namely, Sequence ontology (SO) [142], the National Cancer Institute (NCI) thesaurus [143], Parasite Life Cycle ontology (PL) [144], and the W3C OWL Time ontology [145].

The SO models biological sequences and is a joint effort by genome annotation centers and users of sequence annotation data [142]. The PE ontology re-uses multiple SO classes, including `so:plasmid`, `so:genome` along with its subclasses such as `so:chromosome`, `so:gene`, and `so:flanking_region`. Similarly, `NCI:gene_function`, `PL:organism_strain`, `Time:DateTimeDescription` are some of the other classes re-used in PE ontology from the NCI, PL, and OWL Time ontology respectively. In addition, PE ontology also re-uses the object property `PL:has_base_strain` from PL ontology. Therefore, the PE ontology not only allows interoperability with domain-specific provenance ontologies by extending the Provenir ontology, but also ensures interoperability with existing biomedical ontologies listed at NCBO.

7.4 Query Infrastructure of *T.cruzi* PMS

In contrast to the existing informatics infrastructure in the Tarleton research group, the *T.cruzi* PMS uses the provenance query operators (implemented in a query engine) to execute provenance queries. Given an input value, the query operators compose the corresponding SPARQL query pattern. Figure 7-4 describes

the use of the *provenance ()* query operator to answer the example provenance queries introduced in Section 7.1.2.

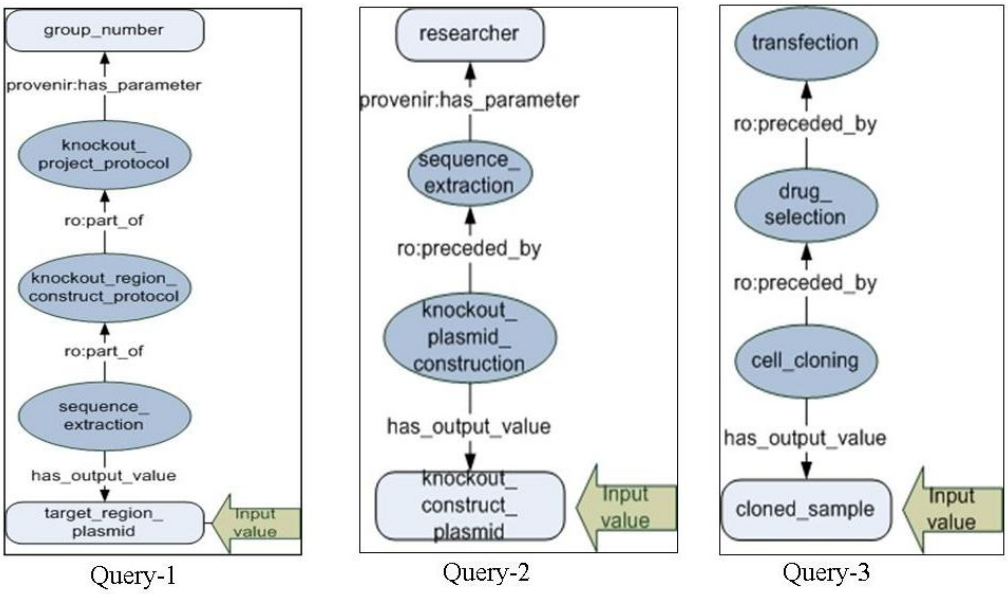


Figure 7-4 Use of *provenance ()* query operator to answer example provenance queries

Provenance queries are path computations over RDF graphs and are expensive operations that require computation of fixed paths, recursive pattern-based paths and neighborhood retrieval. As discussed in [146], a straightforward implementation does not scale with the large scale datasets for complex provenance queries, hence a new class of materialized views called materialized provenance views (MPV) have been defined in PMF [146].

Theoretically, the MPV correspond to a single logical unit of provenance in a given domain, for example one complete experiment cycle in *T.cruzi* domain. A logical unit of provenance information is identified using the domain-specific ontology used for an application. The MPV in *T.cruzi* PMS is defined using the PE ontology as a set of processes starting with the `sequence_extraction` class and terminating with the `cell_cloning` class (Figure 7-5). An important advantage of defining an MPV using the PE ontology is the ability of a single MPV to satisfy all queries for data entities created or used in a single experiment cycle.

The query engine uses a B-tree index to identify query inputs that can be satisfied by a MPV instead of being executed against the underlying database. The use of MPV results in significant gain in query response time with increasing data size and complexity of provenance query expression pattern. The next section discusses the evaluation results of the provenance queries.

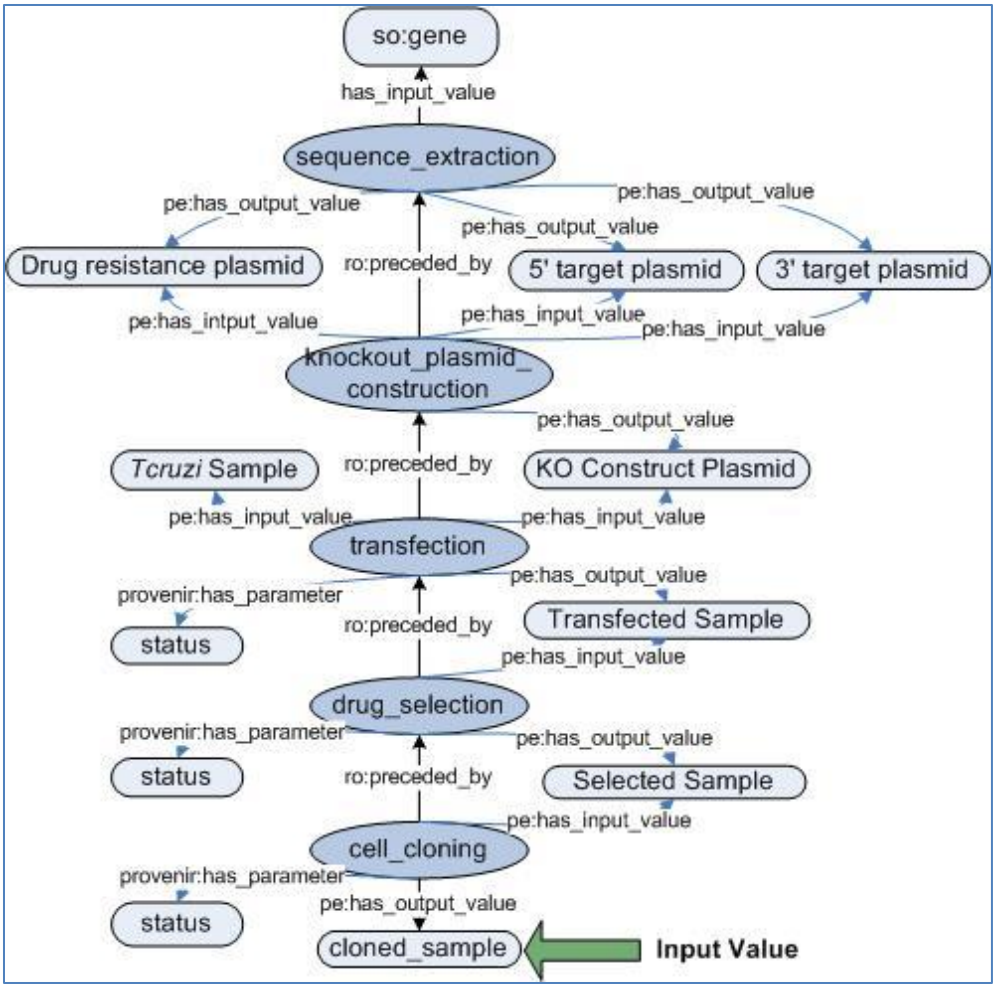


Figure 7-5 The result of Query 4 corresponds to a MPV in *T.cruzi* provenance system

7.5 Evaluation

The objective of our evaluation of the *T.cruzi* PMS is three-fold:

1. Verify that the example provenance queries (Section 7.1.2) can be answered correctly in the *T.cruzi* PMS
2. Evaluate the scalability of *T.cruzi* PMS with increasing size of RDF data

Evaluate the ability of the *T.cruzi* PMS to answer increasingly complex provenance queries.

7.5.1 Experiment Setup, Queries, and Dataset

The experiments were conducted using Oracle10g (Release 10.2.0.3.0) DBMS on a Sun Fire V490 server running 64-bit Solaris 9 with four 1.8 GHz Ultra Sparc IV processors and 8GB of main memory. The database used an 8 KB block size and was configured with a 512 MB buffer cache.

The dataset (Table 7.1) corresponds to a number of experiment cycles and were generated in the Tarleton research group. The standard RDFS entailment rules and two user defined rules were used to create new inferred triples (Table 7.2). The first user-defined rule asserts that "If the input value of a process (p1) is same as output value of another process (p2), then p1 is linked to p2 by property `proevnir:preceded_by`". The second user-defined rule asserts that "If a process (p1) is part of another process (p2) and pa1 is a parameter for p2, then pa1 is also a parameter for process (p1).

Table 7.1The four RDF datasets used to evaluate scalability of *T.cruzi* provenance system

Dataset ID	Number of RDF Inferred Triples	Total Number of RDF Triples
DS 1	2,673	3,553
DS 2	3,470	4,490
DS 3	4,988	6,288
DS 4	47,133	60,912

The SPARQL query patterns corresponding to the example provenance queries represent varying levels of query patterns complexity in terms of total number of variables, the total number of triples, and use of the SPARQL OPTIONAL function [117]. This complexity is also called "expression complexity" [116], and Table 7.2 lists the expression complexity of the example queries expressed in SPARQL syntax.

Table 7.2 Details of SPARQL queries with increasing expression complexity

Query ID	Number of Variables	Total Number of Triples	Nesting Levels using OPTIONAL
Query 1: Target plasmid	25	84	4
Query 2: Plasmid_66	38	110	5
Query 3: Transfection attempts	67	190	7
Query 4: cloned_sample66	67	190	7

7.5.2 Experiment 1

This experiment involved the verification of the results for the four queries executed using the T.cruzi PMS by the informatics specialist in the Tarleton research group. The results of the four queries are:

1) *"Group1" used "target_region_plasmid_Tc00.1047053504033.170_1" target region plasmid to create cloned samples*

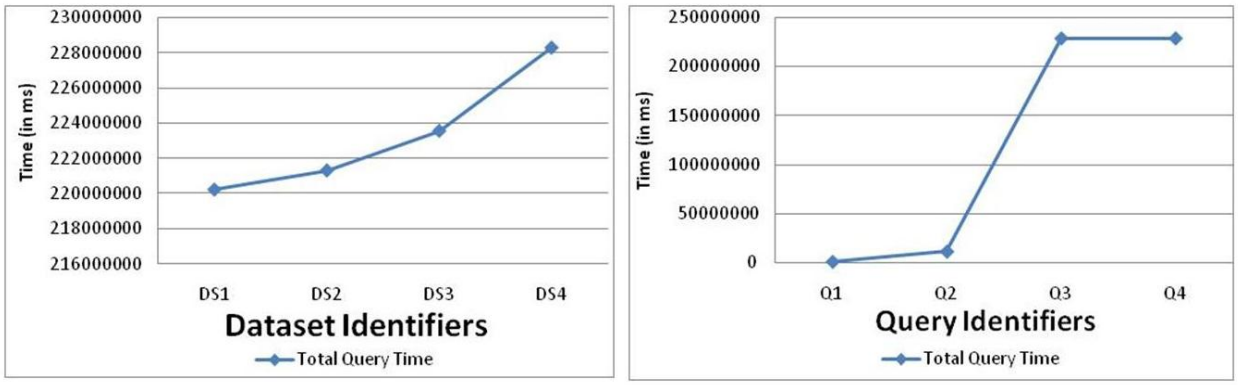
2) *Researcher with user ID = "1" created the knockout plasmid "plasmid66"*

3) "*Cloned sample 66", which is not episomal, involved 1 transfection attempt.*

4) *Gene with identifier "Tc00.1047053506727.100" was used create the cloned sample "cloned_sample66".*

7.5.3 Experiment 2

The four queries, Q1 to Q4 (in Table 7.2), were executed against a fixed RDF dataset, DS4 (in Table 7.1) to evaluate the performance of query engine for provenance queries with increasing complexity. Figure 7-6(a) shows that the response time increases with increasing complexity of the provenance queries. Similarly, to evaluate the performance of the query engine with increasing size of data, query Q4 (in Table 7.2) is executed against the four RDF datasets, DS1 to DS4 (in Table 7.1). Figure 7-6(b) shows that the response time of the query engine increases with increasing size of RDF data.



(a) (b)

Figure 7-6 The response time for provenance query engine with (a) increasing size of RDF dataset and (b) increasing complexity of queries

The two sets of results demonstrate the need for effective optimization techniques to enable practical use of the query engine in the *T.cruzi* PMS. The next experiment discusses the results of using the MPV for query optimization.

7.5.4 Experiment 3

Using the results of "Experiment 2" as baseline, this experiment discusses the significant improvement in response time of the provenance query engine using MPV. Figure 7-7(a) shows the comparative results for provenance queries with increasing complexity executed against the underlying database and the MPV. Similar to "Experiment 2", Figure 7-7(a) describes the result of executing the four queries (in Table 7.2) using the fixed dataset DS 4 (in Table 7.1). The MPV used in Figure 7-7(a) corresponds to the provenance result of "*Cloned sample 66*" consisting of 139 RDF triples and occupying 27KB space. We

note that this single MPV is used to answer all the four queries, Q1 to Q4 (in Table 7.2). Figure 7-7(b) shows the benefit of using MPV for query Q4 (in Table 7.2) over increasing size of RDF datasets, DS1 to DS4 (in Table 7.1).

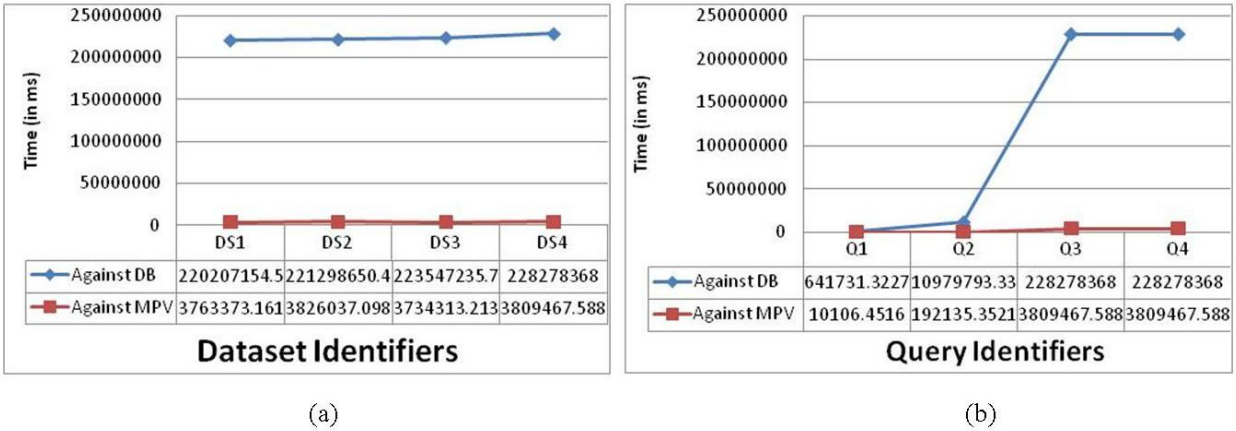


Figure 7-7 Comparative results for (a) increasing size of RDF datasets over the underlying database and MPV and (b) provenance queries with increasing complexity

The results demonstrate that the use of MPV leads to significant improvement in response time, for both increasing complexity of provenance queries and increasing size of RDF dataset.

7.6 Related Work

Provenance has been studied in both the eScience [38] and the database community [58]. In the eScience community, provenance management has been addressed primarily in the context of workflow engines [95] [88], but recent work has argued for use of domain semantics in eScience provenance [5]. Simmhan et al. [38] survey the provenance management issues in eScience. The database community has also addressed the issue of provenance and defined various types of provenance, for example "why provenance" [36] and "where provenance" [36]. Database provenance is also described as fine-grained provenance [58]. A detailed comparison of PMF (that underpins the *T.cruzi* PMS) with both workflow and database provenance is presented in [146].

The Semantic Provenance Capture in Data Ingest Systems (SPCDIS) [147] is an example of eScience project with dedicated infrastructure for provenance management. In contrast to the *T.cruzi* PMS, the SPCDIS project uses the proof markup language (PML) [148] to capture provenance information. The Inference Web toolkit [148] features a set of tools to generate, register and search proofs encoded in PML. Both *T.cruzi* PMS and the SPCDIS have common objectives but use different approaches to achieve them, specifically the *T.cruzi* PMS uses an ontology-driven approach with robust query infrastructure for provenance management.

7.7 Conclusions

This chapter introduces an in-use ontology-driven provenance management infrastructure for parasite research called *T.cruzi* PMS. The following conclusions are drawn from our experience described in this paper:

1. Domain-specific provenance ontologies, such as PE ontology, are the key component for an eScience provenance management infrastructure. Further, by extending the *provenir* ontology, the PE ontology can interoperate with other domain-specific provenance ontologies to facilitate sharing and integration of provenance information from different domains and projects.
2. The provenance query operators effectively support provenance queries and provide the users with a well-defined and robust mechanism to execute complex provenance queries.
3. The *T.cruzi* PMS, using MPV-based query optimization, is a scalable infrastructure for increasing data size as well as complexity of provenance queries.

In future, we plan to integrate other experiment protocols in the Tarleton research group such as proteome analysis and sample characterization in the *T.cruzi* PMS.

Chapter 8

Conclusions and Future Work

The eScience paradigm is creating a deluge of scientific data and the role of provenance metadata is gaining critical importance in management, analysis, and associating trust with the datasets. The number of research projects, special interest groups (for example, the W3C Provenance Incubator Group), special issues in the journals etc. demonstrate the burgeoning interest and work in provenance related research. Traditionally, provenance has been studied in the context of relational databases in computer sciences, but the rapid adoption of scientific workflows to automate scientific processes has resulted in work on workflow provenance.

In this dissertation, we identified the shortcomings of database and workflow provenance in addressing the unique set of challenges presented by eScience that requires the use of "domain semantics" in management of provenance information. To address this challenge we introduced "Semantic Provenance" that not only incorporates domain semantics in provenance models, but also with its well-defined formal semantics enables software applications to correctly interpret large volumes of scientific data consistently. We identified three aspects of Semantic Provenance that need to defined to facilitate the use of the Semantic Provenance framework in real world eScience applications, namely (1) an expressive, formal, and extensible model of provenance that can be used as reference model in different scientific domains thereby facilitating provenance interoperability, (2) a mechanism to support complex user defined provenance queries that also does not require the users to learn a specific query language, and (3) a practical implementation that can scale with increasing size of scientific datasets and complexity of provenance query patterns.

Provenance Representation

To meet the requirement of provenance management in proteomics data analysis using mass spectrometry, we created one of the first provenance ontologies called ProPreO. Using the W3C OWL-DL ontology language, ProPreO ontology modeled the comprehensive provenance information associated with proteomics data analysis including, the experiment datasets, the analytical instruments, and the experimental conditions in which the datasets were generated. ProPreO ontology with 490 classes, 35 named relationships, and 3.1 million instance values was released for public use through the NCBO.

We realized that extending the ProPreO ontology to model provenance information in other scientific domain was not practical; hence as presented in Chapter 3 we defined a modular approach for creating inter-operable domain-specific provenance ontologies using an upper-level provenance ontology as

reference. The Provenir upper-level provenance ontology was defined using primitive ontological concepts from the abstract Basic Formal Ontology and models a minimal set of provenance terms that were common across multiple scientific domains. The Provenir ontology, similar to other upper-level ontologies, enables consistent design principles, facilitates domain ontology interoperability and integration in provenance representation. We validated the flexibility and extensibility of the Provenir ontology by creating three domain-specific provenance ontologies in biomedicine (Parasite Experiment ontology), scientific workflows (Janus ontology), and oceanography (Trident ontology) domains by extending the Provenir ontology.

Provenance Query

In addition to provenance representation, we also addressed the issue of querying provenance data. To create a provenance query infrastructure, we first classified the variety of provenance queries into four categories in Chapter 4. This classification scheme used the input and output values of the provenance queries to distinguish between the different categories of queries, namely (a) queries to retrieve the provenance information associated with an entity, (b) queries to retrieve entities that satisfy a set of constraints defined on the provenance information, (c) queries that compare the provenance information associated with two or more entities to interpret the entities to have been created under equivalent, similar, or dissimilar conditions, and (d) queries to combine two or more provenance traces associated with an entity.

We used the query classification scheme to define a set of provenance query operators that creates a well-defined and easy-to-use mechanism (without having to manually compose complex query patterns) for domain users to query provenance information. The four provenance query operators (a) *provenance ()*, (b) *provenance_context ()*, (c) *provenance_compare ()*, and (d) *provenance_merge ()* supports provenance queries in each of the four categories defined in the classification scheme.

In Chapter 5, we implemented a provenance query engine over a RDF data store to support the provenance query operators using the SPARQL RDF query language. Using the standard approach to evaluate new query operators, we used a set of increasingly complex provenance query patterns and large size of RDF datasets (the largest dataset with more than 308 million RDF triples) to test the query expression and data complexity of the provenance query operators. We found that a straightforward implementation of the provenance query engine, using the Oracle 10g RDF data store, does not scale and we defined a novel class of materialized views using the Provenir ontology called Materialized Provenance Views (MPV) for query optimization. The MPVs are defined in terms of the Provenir ontology that allows the provenance query engine to automatically compute the relevant MPV for different application domains using the standard RDFS entailment rules.

Provenance Tracking in Semantic Web Applications

The use of RDF model to store and disseminate scientific datasets is rapidly gaining momentum in the scientific community with several large databases such as UniprotDB, KEGG, and projects such as the Biomedical Knowledge Repository at the US National Library of Medicine. An important challenge in these projects is to efficiently keep track of the provenance information associated with the data values represented in RDF. In Chapter 6, we identified the drawbacks associated with the RDF reification approach and proposed the Provenance Context Entity (PaCE) approach as an efficient mechanism to track provenance of RDF triples.

The PaCE approach defines a formal context structure consisting of appropriate provenance information associated with the entities of a RDF triple to accurately interpret the RDF triple. The provenance context is application-specific and is defined using the ontology concepts and relationships of the Provenir ontology or a domain-specific provenance ontology that extends Provenir ontology. The evaluation of the PaCE approach demonstrated that it reduces by 49% the total number of RDF triples required to track provenance in scientific datasets and also improves the query performance by three orders of magnitude for complex provenance queries.

A Framework for Provenance Management in Translational Medicine Research

In Chapter 7, we discussed the application of the Semantic Provenance framework in an example translational research project to identify vaccine targets for the human pathogen *T.cruzi* that affects more than 12 million people in Latin America. We demonstrated the use of the provenance system to address the different stages of provenance information lifecycle in the *T.cruzi* SPSE namely, (a) provenance capture, (b) provenance representation, (c) provenance storage, and (d) provenance querying. The provenance system includes a domain-specific provenance ontology called the Parasite Experiment Ontology (PEO) that was created to model provenance information associated with the experiment protocols used in the Tarleton research group by extending the Provenir ontology. The evaluation of the provenance system validated the use of the Semantic Provenance framework in real world applications.

Executive Summary

We conclude that use of provenance metadata to manage the growing deluge of scientific data in the eScience paradigm requires us to address multiple aspects of provenance management. In this dissertation, we defined a new category of provenance called Semantic Provenance and addressed the associated challenges of provenance representation, through creation of the Provenir upper-level provenance ontology, and provenance query mechanism, by defining provenance query operators and implementing a highly scalable query engine, to create practical provenance systems for real world applications in eScience.

8.1 Future Work

The use of Provenir ontology in an increasing number of scientific domains, including ongoing projects in Semantic Sensor Web (SSW) and patient health care, addresses the issue of a common model for provenance representation to facilitate provenance interoperability. In case of provenance query infrastructure, we believe that specialized query operators can be defined by extending the four provenance query operators introduced in this thesis (three examples of the specialized query operators are also introduced in Chapter 4). The other aspect of provenance query is the challenge of optimizing provenance queries. In this dissertation, we used materialization informed by the Provenir ontology for query optimization, but the definition of indexing structure that take into consideration the characteristics of the provenance queries are an exciting area of research. Recent index-based provenance query optimization approaches have been largely in the context of relational database provenance or XML. We believe new indexing approaches need to be defined for provenance information represented in RDF. Further, we did not explore the use of SPARQL query optimization techniques in this dissertation and we plan to explore this in future.

The use of provenance information as a foundational layer to create trust models or compute trust values is another area of exciting research. Current trust models often use arbitrary values to initialize trust computation, but we believe that use of provenance information to bootstrap computation of trust values in conjunction of trust models will significantly influence the quality of the trust values. The close integration of the trust and provenance layers in has significant implications for multiple scientific domains including sensor networks, biomedicine, eGovernance, library information systems, and clinical informatics.

Appendix A

Provenance Query Patterns in SPARQL Syntax

The queries below are the first and fifth provenance queries expressed in SPARQL syntax used in Chapter 5 to evaluate the expression complexity of the *provenance ()* query operator.

Query 5:

```
PREFIX  provenir: <http://knoesis.wright.edu/provenir/provenir.owl#>
PREFIX  trident: <http://www.trident.owl/trident#>
PREFIX  rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

CONSTRUCT
 { ?p1 provenir:has_participant trident:ChartDataTable90829493849637 .
   ?p1 rdf:type provenir:process .
   ?p1 provenir:has_parameter ?pa1 .
   ?pa1 rdf:type provenir:parameter .
   ?p1 provenir:part_of ?p2 .
   ?p1 rdf:type provenir:process .
   ?p2 rdf:type provenir:process .
   ?p1 provenir:has_agent ?a1 .
   ?p1 rdf:type provenir:process .
   ?a1 rdf:type provenir:agent .
   ?a1 provenir:has_parameter ?pa2 .
   ?pa2 rdf:type provenir:parameter .
   ?a1 provenir:part_of ?a2 .
   ?a1 rdf:type provenir:agent .
   ?a2 rdf:type provenir:agent .
   ?a1 provenir:contained_in ?a3 .
   ?a1 rdf:type provenir:agent .
   ?a3 rdf:type provenir:agent .
   ?a1 provenir:adjacent_to ?a4 .
   ?a1 rdf:type provenir:agent .
   ?a4 rdf:type provenir:agent .
   ?a5 provenir:part_of ?a1 .
   ?a5 rdf:type provenir:agent .
   ?a1 rdf:type provenir:agent .
   ?a6 provenir:contained_in ?a1 .
   ?a6 rdf:type provenir:agent .
   ?a1 rdf:type provenir:agent .
   ?a7 provenir:adjacent_to ?a1 .
```

?a7 rdf:type provenir:agent .
?a1 rdf:type provenir:agent .
?p1 provenir:preceded_by ?p3 .
?p1 rdf:type provenir:process .
?p3 rdf:type provenir:process .
?p3 provenir:has_parameter ?pa3 .
?pa3 rdf:type provenir:parameter .
?p3 provenir:part_of ?p4 .
?p3 rdf:type provenir:process .
?p4 rdf:type provenir:process .
?p3 provenir:has_participant ?d1 .
?p3 rdf:type provenir:process .
?d1 rdf:type provenir:data_collection .
?d1 provenir:derives_from ?d2 .
?d1 rdf:type provenir:data_collection .
?d2 rdf:type provenir:data_collection .
?d1 provenir:transformation_of ?d3 .
?d1 rdf:type provenir:data_collection .
?d3 rdf:type provenir:data_collection .
?p3 provenir:has_agent ?a8 .
?p3 rdf:type provenir:process .
?a8 rdf:type provenir:agent .
?a8 provenir:has_parameter ?pa4 .
?pa4 rdf:type provenir:parameter .
?a8 provenir:part_of ?a9 .
?a8 rdf:type provenir:agent .
?a9 rdf:type provenir:agent .
?a8 provenir:contained_in ?a10 .
?a8 rdf:type provenir:agent .
?a10 rdf:type provenir:agent .
?a8 provenir:adjacent_to ?a11 .
?a8 rdf:type provenir:agent .
?a11 rdf:type provenir:agent .
?a12 provenir:part_of ?a8 .
?a12 rdf:type provenir:agent .
?a8 rdf:type provenir:agent .
?a13 provenir:contained_in ?a8 .
?a13 rdf:type provenir:agent .
?a8 rdf:type provenir:agent .
?a14 provenir:adjacent_to ?a8 .
?a14 rdf:type provenir:agent .
?a8 rdf:type provenir:agent .
?p3 provenir:preceded_by ?p5 .

?p3 rdf:type provenir:process .
?p5 rdf:type provenir:process .
?p5 provenir:has_parameter ?pa5 .
?pa5 rdf:type provenir:parameter .
?p5 provenir:part_of ?p6 .
?p5 rdf:type provenir:process .
?p6 rdf:type provenir:process .
?p5 provenir:has_participant ?d4 .
?p5 rdf:type provenir:process .
?d4 rdf:type provenir:data_collection .
?d4 provenir:derives_from ?d5 .
?d4 rdf:type provenir:data_collection .
?d5 rdf:type provenir:data_collection .
?d4 provenir:transformation_of ?d6 .
?d4 rdf:type provenir:data_collection .
?d6 rdf:type provenir:data_collection .
?p5 provenir:has_agent ?a15 .
?p5 rdf:type provenir:process .
?a15 rdf:type provenir:agent .
?a15 provenir:has_parameter ?pa6 .
?pa6 rdf:type provenir:parameter .
?a15 provenir:part_of ?a16 .
?a15 rdf:type provenir:agent .
?a16 rdf:type provenir:agent .
?a15 provenir:contained_in ?a17 .
?a15 rdf:type provenir:agent .
?a17 rdf:type provenir:agent .
?a15 provenir:adjacent_to ?a18 .
?a15 rdf:type provenir:agent .
?a18 rdf:type provenir:agent .
?a19 provenir:part_of ?a15 .
?a19 rdf:type provenir:agent .
?a15 rdf:type provenir:agent .
?a20 provenir:contained_in ?a15 .
?a20 rdf:type provenir:agent .
?a15 rdf:type provenir:agent .
?a21 provenir:adjacent_to ?a15 .
?a21 rdf:type provenir:agent .
?a15 rdf:type provenir:agent .
?p5 provenir:preceded_by ?p7 .
?p5 rdf:type provenir:process .
?p7 rdf:type provenir:process .
?p7 provenir:has_parameter ?pa7 .

?pa7 rdf:type provenir:parameter .
?p7 provenir:part_of ?p8 .
?p7 rdf:type provenir:process .
?p8 rdf:type provenir:process .
?p7 provenir:has_participant ?d7 .
?p7 rdf:type provenir:process .
?d7 rdf:type provenir:data_collection .
?d7 provenir:derives_from ?d8 .
?d7 rdf:type provenir:data_collection .
?d8 rdf:type provenir:data_collection .
?d7 provenir:transformation_of ?d9 .
?d7 rdf:type provenir:data_collection .
?d9 rdf:type provenir:data_collection .
?p7 provenir:has_agent ?a22 .
?p7 rdf:type provenir:process .
?a22 rdf:type provenir:agent .
?a22 provenir:has_parameter ?pa8 .
?pa8 rdf:type provenir:parameter .
?a22 provenir:part_of ?a23 .
?a22 rdf:type provenir:agent .
?a23 rdf:type provenir:agent .
?a22 provenir:contained_in ?a24 .
?a22 rdf:type provenir:agent .
?a24 rdf:type provenir:agent .
?a22 provenir:adjacent_to ?a25 .
?a22 rdf:type provenir:agent .
?a25 rdf:type provenir:agent .
?a26 provenir:part_of ?a22 .
?a26 rdf:type provenir:agent .
?a22 rdf:type provenir:agent .
?a27 provenir:contained_in ?a22 .
?a27 rdf:type provenir:agent .
?a22 rdf:type provenir:agent .
?a28 provenir:adjacent_to ?a22 .
?a28 rdf:type provenir:agent .
?a22 rdf:type provenir:agent .
?p7 provenir:preceded_by ?p9 .
?p7 rdf:type provenir:process .
?p9 rdf:type provenir:process .
?p9 provenir:has_parameter ?pa9 .
?pa9 rdf:type provenir:parameter .
?p9 provenir:part_of ?p10 .
?p9 rdf:type provenir:process .

```
   ?p10 rdf:type provenir:process .
   ?p9 provenir:has_participant ?d10 .
   ?p9 rdf:type provenir:process .
   ?d10 rdf:type provenir:data_collection .
   ?d10 provenir:derives_from ?d11 .
   ?d10 rdf:type provenir:data_collection .
   ?d11 rdf:type provenir:data_collection .
   ?d10 provenir:transformation_of ?d12 .
   ?d10 rdf:type provenir:data_collection .
   ?d12 rdf:type provenir:data_collection .
   ?p9 provenir:has_agent ?a29 .
   ?p9 rdf:type provenir:process .
   ?a29 rdf:type provenir:agent .
   ?a29 provenir:has_parameter ?pa10 .
   ?pa10 rdf:type provenir:parameter .
   ?a29 provenir:part_of ?a30 .
   ?a29 rdf:type provenir:agent .
   ?a30 rdf:type provenir:agent .
   ?a29 provenir:contained_in ?a31 .
   ?a29 rdf:type provenir:agent .
   ?a31 rdf:type provenir:agent .
   ?a29 provenir:adjacent_to ?a32 .
   ?a29 rdf:type provenir:agent .
   ?a32 rdf:type provenir:agent .
   ?a33 provenir:part_of ?a29 .
   ?a33 rdf:type provenir:agent .
   ?a29 rdf:type provenir:agent .
   ?a34 provenir:contained_in ?a29 .
   ?a34 rdf:type provenir:agent .
   ?a29 rdf:type provenir:agent .
   ?a35 provenir:adjacent_to ?a29 .
   ?a35 rdf:type provenir:agent .
   ?a29 rdf:type provenir:agent .}
WHERE
 { OPTIONAL
    { ?p1  provenir:has_participant  trident:ChartDataTable90829493849637 ;
         rdf:type          provenir:process .
      OPTIONAL
       { ?p1   provenir:has_parameter  ?pa1 .
        ?pa1  rdf:type           provenir:parameter .
       }
      OPTIONAL
       { ?p1  provenir:part_of  ?p2 ;
```

```
      rdf:type    provenir:process .
  ?p2  rdf:type    provenir:process .
 }
OPTIONAL
 { ?p1  provenir:has_agent  ?a1 ;
      rdf:type     provenir:process .
  ?a1  rdf:type     provenir:agent .
  OPTIONAL
   { ?a1   provenir:has_parameter  ?pa2 .
     ?pa2  rdf:type           provenir:parameter .
   }
  OPTIONAL
   { ?a1  provenir:part_of  ?a2 ;
        rdf:type    provenir:agent .
     ?a2  rdf:type    provenir:agent .
   }
  OPTIONAL
   { ?a1  provenir:contained_in  ?a3 ;
        rdf:type       provenir:agent .
     ?a3  rdf:type       provenir:agent .
   }
  OPTIONAL
   { ?a1  provenir:adjacent_to  ?a4 ;
        rdf:type      provenir:agent .
     ?a4  rdf:type      provenir:agent .
   }
  OPTIONAL
   { ?a5  provenir:part_of  ?a1 ;
        rdf:type    provenir:agent .
     ?a1  rdf:type    provenir:agent .
   }
  OPTIONAL
   { ?a6  provenir:contained_in  ?a1 ;
        rdf:type       provenir:agent .
     ?a1  rdf:type       provenir:agent .
   }
  OPTIONAL
   { ?a7  provenir:adjacent_to  ?a1 ;
        rdf:type      provenir:agent .
     ?a1  rdf:type      provenir:agent .
   }
 }
OPTIONAL
```

```
{ ?p1  provenir:preceded_by  ?p3 ;
     rdf:type       provenir:process .
 ?p3  rdf:type       provenir:process .
 OPTIONAL
  { ?p3   provenir:has_parameter  ?pa3 .
    ?pa3  rdf:type           provenir:parameter .
  }
 OPTIONAL
  { ?p3  provenir:part_of  ?p4 ;
       rdf:type    provenir:process .
    ?p4  rdf:type    provenir:process .
  }
 OPTIONAL
  { ?p3  provenir:has_participant  ?d1 ;
       rdf:type          provenir:process .
    ?d1  rdf:type          provenir:data_collection .
    OPTIONAL
     { ?d1  provenir:derives_from  ?d2 ;
          rdf:type        provenir:data_collection .
       ?d2  rdf:type        provenir:data_collection .
     }
    OPTIONAL
     { ?d1  provenir:transformation_of  ?d3 ;
          rdf:type           provenir:data_collection .
       ?d3  rdf:type           provenir:data_collection .
     }
  }
 OPTIONAL
  { ?p3  provenir:has_agent  ?a8 ;
       rdf:type     provenir:process .
    ?a8  rdf:type     provenir:agent .
    OPTIONAL
     { ?a8   provenir:has_parameter  ?pa4 .
       ?pa4  rdf:type           provenir:parameter .
     }
    OPTIONAL
     { ?a8  provenir:part_of  ?a9 ;
          rdf:type    provenir:agent .
       ?a9  rdf:type    provenir:agent .
     }
    OPTIONAL
     { ?a8   provenir:contained_in  ?a10 ;
          rdf:type        provenir:agent .
```

```
    ?a10 rdf:type       provenir:agent .
   }
  OPTIONAL
   { ?a8  provenir:adjacent_to  ?a11 ;
        rdf:type      provenir:agent .
    ?a11 rdf:type      provenir:agent .
   }
  OPTIONAL
   { ?a12 provenir:part_of ?a8 ;
        rdf:type   provenir:agent .
    ?a8  rdf:type   provenir:agent .
   }
  OPTIONAL
   { ?a13 provenir:contained_in ?a8 ;
        rdf:type       provenir:agent .
    ?a8  rdf:type       provenir:agent .
   }
  OPTIONAL
   { ?a14 provenir:adjacent_to ?a8 ;
        rdf:type      provenir:agent .
    ?a8  rdf:type      provenir:agent .
   }
 }
OPTIONAL
 { ?p3 provenir:preceded_by ?p5 ;
     rdf:type      provenir:process .
  ?p5 rdf:type      provenir:process .
  OPTIONAL
   { ?p5  provenir:has_parameter ?pa5 .
    ?pa5 rdf:type          provenir:parameter .
   }
  OPTIONAL
   { ?p5 provenir:part_of ?p6 ;
        rdf:type   provenir:process .
    ?p6 rdf:type   provenir:process .
   }
  OPTIONAL
   { ?p5 provenir:has_participant ?d4 ;
        rdf:type         provenir:process .
    ?d4 rdf:type         provenir:data_collection .
    OPTIONAL
     { ?d4 provenir:derives_from ?d5 ;
          rdf:type       provenir:data_collection .
```

```
    ?d5 rdf:type       provenir:data_collection .
   }
  OPTIONAL
   { ?d4 provenir:transformation_of ?d6 ;
       rdf:type          provenir:data_collection .
    ?d6 rdf:type          provenir:data_collection .
   }
 }
OPTIONAL
 { ?p5  provenir:has_agent ?a15 ;
     rdf:type    provenir:process .
  ?a15 rdf:type    provenir:agent .
  OPTIONAL
   { ?a15 provenir:has_parameter ?pa6 .
    ?pa6 rdf:type          provenir:parameter .
   }
  OPTIONAL
   { ?a15 provenir:part_of ?a16 ;
       rdf:type   provenir:agent .
    ?a16 rdf:type   provenir:agent .
   }
  OPTIONAL
   { ?a15 provenir:contained_in ?a17 ;
       rdf:type       provenir:agent .
    ?a17 rdf:type       provenir:agent .
   }
  OPTIONAL
   { ?a15 provenir:adjacent_to ?a18 ;
       rdf:type      provenir:agent .
    ?a18 rdf:type      provenir:agent .
   }
  OPTIONAL
   { ?a19 provenir:part_of ?a15 ;
       rdf:type   provenir:agent .
    ?a15 rdf:type   provenir:agent .
   }
  OPTIONAL
   { ?a20 provenir:contained_in ?a15 ;
       rdf:type       provenir:agent .
    ?a15 rdf:type       provenir:agent .
   }
  OPTIONAL
   { ?a21 provenir:adjacent_to ?a15 ;
```

```
        rdf:type       provenir:agent .
    ?a15  rdf:type       provenir:agent .
   }
 }
OPTIONAL
 { ?p5  provenir:preceded_by  ?p7 ;
     rdf:type       provenir:process .
  ?p7  rdf:type       provenir:process .
  OPTIONAL
   { ?p7   provenir:has_parameter  ?pa7 .
    ?pa7  rdf:type           provenir:parameter .
   }
  OPTIONAL
   { ?p7  provenir:part_of  ?p8 ;
       rdf:type    provenir:process .
    ?p8  rdf:type    provenir:process .
   }
  OPTIONAL
   { ?p7  provenir:has_participant  ?d7 ;
       rdf:type          provenir:process .
    ?d7  rdf:type          provenir:data_collection .
    OPTIONAL
     { ?d7  provenir:derives_from  ?d8 ;
         rdf:type       provenir:data_collection .
      ?d8  rdf:type       provenir:data_collection .
     }
    OPTIONAL
     { ?d7  provenir:transformation_of  ?d9 ;
         rdf:type           provenir:data_collection .
      ?d9  rdf:type           provenir:data_collection .
     }
   }
  OPTIONAL
   { ?p7   provenir:has_agent  ?a22 ;
       rdf:type     provenir:process .
    ?a22  rdf:type     provenir:agent .
    OPTIONAL
     { ?a22  provenir:has_parameter  ?pa8 .
      ?pa8  rdf:type           provenir:parameter .
     }
    OPTIONAL
     { ?a22  provenir:part_of  ?a23 ;
         rdf:type    provenir:agent .
```

```
    ?a23 rdf:type   provenir:agent .
   }
  OPTIONAL
   { ?a22 provenir:contained_in ?a24 ;
        rdf:type       provenir:agent .
     ?a24 rdf:type       provenir:agent .
   }
  OPTIONAL
   { ?a22 provenir:adjacent_to ?a25 ;
        rdf:type      provenir:agent .
     ?a25 rdf:type      provenir:agent .
   }
  OPTIONAL
   { ?a26 provenir:part_of ?a22 ;
        rdf:type   provenir:agent .
     ?a22 rdf:type   provenir:agent .
   }
  OPTIONAL
   { ?a27 provenir:contained_in ?a22 ;
        rdf:type       provenir:agent .
     ?a22 rdf:type       provenir:agent .
   }
  OPTIONAL
   { ?a28 provenir:adjacent_to ?a22 ;
        rdf:type      provenir:agent .
     ?a22 rdf:type      provenir:agent .
   }
 }
OPTIONAL
 { ?p7 provenir:preceded_by ?p9 ;
     rdf:type      provenir:process .
  ?p9 rdf:type      provenir:process .
  OPTIONAL
   { ?p9  provenir:has_parameter ?pa9 .
    ?pa9 rdf:type          provenir:parameter .
   }
  OPTIONAL
   { ?p9  provenir:part_of ?p10 ;
        rdf:type   provenir:process .
    ?p10 rdf:type   provenir:process .
   }
  OPTIONAL
   { ?p9  provenir:has_participant ?d10 ;
```

```
      rdf:type          provenir:process .
  ?d10  rdf:type          provenir:data_collection .
  OPTIONAL
   { ?d10  provenir:derives_from  ?d11 ;
        rdf:type        provenir:data_collection .
     ?d11  rdf:type        provenir:data_collection .
   }
  OPTIONAL
   { ?d10  provenir:transformation_of  ?d12 ;
        rdf:type          provenir:data_collection .
     ?d12  rdf:type          provenir:data_collection .
   }
 }
OPTIONAL
 { ?p9   provenir:has_agent  ?a29 ;
      rdf:type     provenir:process .
  ?a29  rdf:type     provenir:agent .
  OPTIONAL
   { ?a29   provenir:has_parameter  ?pa10 .
     ?pa10  rdf:type          provenir:parameter .
   }
  OPTIONAL
   { ?a29  provenir:part_of  ?a30 ;
        rdf:type    provenir:agent .
     ?a30  rdf:type    provenir:agent .
   }
  OPTIONAL
   { ?a29  provenir:contained_in  ?a31 ;
        rdf:type        provenir:agent .
     ?a31  rdf:type        provenir:agent .
   }
  OPTIONAL
   { ?a29  provenir:adjacent_to  ?a32 ;
        rdf:type       provenir:agent .
     ?a32  rdf:type       provenir:agent .
   }
  OPTIONAL
   { ?a33  provenir:part_of  ?a29 ;
        rdf:type    provenir:agent .
     ?a29  rdf:type    provenir:agent .
   }
  OPTIONAL
   { ?a34  provenir:contained_in  ?a29 ;
```

```
                    rdf:type        provenir:agent .
                  ?a29  rdf:type        provenir:agent .
                 }
                OPTIONAL
                 { ?a35  provenir:adjacent_to  ?a29 ;
                    rdf:type       provenir:agent .
                  ?a29  rdf:type       provenir:agent .
                 }
               }
             }
           }
         }
       }
  }
 OPTIONAL
  { trident:ChartDataTable90829493849637
        provenir:derives_from  ?d13 .
   ?d13     rdf:type        provenir:data_collection .
  }
 OPTIONAL
  { trident:ChartDataTable90829493849637
        provenir:contained_in  ?d14 .
   ?d14     rdf:type        provenir:data_collection .
  }
 OPTIONAL
  { ?d15  provenir:contained_in  trident:ChartDataTable90829493849637 ;
      rdf:type        provenir:data_collection .
  }
 OPTIONAL
  { trident:ChartDataTable90829493849637
        provenir:part_of  ?d16 .
   ?d16     rdf:type    provenir:data_collection .
  }
 OPTIONAL
  { ?d17  provenir:part_of  trident:ChartDataTable90829493849637 ;
      rdf:type    provenir:data_collection .
  }
 OPTIONAL
  { trident:ChartDataTable90829493849637
        provenir:transformation_of  ?d18 .
   ?d18     rdf:type            provenir:data_collection .
  }
}
```

Bibliography

1. Smalheiser, N.R.: Informatics and hypothesis-driven research. EMBO Reports **3** (2002) 702

2. Hey, T., Trefethen, A. E. : Cyberinfrastructure for e-Science. Science **308** (2005) 817 - 821

3. Cancer Biomedical Informatics Grid (caBIG), https://cabig.nci.nih.gov/, retrieved on July 15, 2010.

4. Vastrik, I., D'Eustachio, P., Schmidt, E., Joshi-Tope, G., Gopinath, G., Croft, D., de Bono, B., Gillespie, M., Jassal, B., Lewis, S., Matthews, L., Wu, G., Birney, E., Stein, L.: Reactome: a knowledge base of biologic pathways and processes. Genome Biology **8** (2007)

5. Sahoo, S.S., Sheth, A., Henson, C.: Semantic Provenance for eScience: Managing the Deluge of Scientific Data. IEEE Internet Computing **12** (2008) 46-54

6. Sheth, A., Klas, W.: Multimedia Data Management: Using Metadata to Integrate and Apply Digital Media. McGraw-Hill (1998)

7. Myers, J.D., Pancerella, C., Lansing, C., Schuchardt, K., Didier, B.: Multi-scale Science: Supporting Emerging Practice with Semantically-Derived Provenance. Semantic Web Technologies for Searching and Retrieving Scientific Data Workshop at the 2nd International Semantic Web Conference (2003)

8. The Library of Congress, www.loc.gov/standards, retrieved on July 15, 2010.

9. OGC GML, www.opengeospatial.org/standards/gml, retrieved on July 15, 2010.

10. Boll, S., Klas, W., Sheth, A.P.: Overview on Using Metadata to manage Multimedia Data. Multimedia Data Management (1998) 1-24

11. The National Center for Biomedical Ontology, http://www.bioontology.org/, retrieved on July 15, 2010.

12. Lassila, O.: Web Metadata: A Matter of Semantics. IEEE Internet Computing **2** (1998) 30-37

13. Sheth, A.P.a.J.A.L.: Federated database systems for managing distributed, heterogeneous and autonomous databases. ACM Computing Surveys **22** (1990) 183--236

14. Kashyap V, S.A.: Information brokering across heterogeneous digital data - a metadata-based approach. Kluwer Academic, Boston, Mass., USA (2000)

15. Kashyap, V., Shah, K., Sheth, A.: Metadata for building the multimedia patch quilt. Multimedia Database Systems: Issues and Research Directions (1996) 297-319

16. McCarthy, J.L.: Metadata Management for Large Statistical Databases. 8th International Conference on Very Large Data Bases (1982) 234-243

17. Kashyap, V., Sheth, A.: Schematic and Semantic Similarities between Database Objects: A Context-based Approach. VLDB Journal **5** (1996) 276–304

18. Böhm, K., Rakow, T. C.: Metadata for multimedia documents. SIGMOD Rec. **23** (1994) 21-26

19. Understanding Metadata. National Information Standards Organization (2004)

20. Sheth, A.: Data Semantics: What, Where, and How? In: Mark, R.M.a.L. (ed.): Data Semantics (IFIP Transactions). Chapman & Hall, London (1996) 601–610

21. Sheth, A.: Semantic Meta Data for Enterprise Information Integration. Data Management Review (2003)

22. Berners-Lee, T., Hendler, J., Lassila, O. : The Semantic Web. Scientific American Magazine. (2001)

23. Bizer, C., Heath, T., Berners-Lee, T.: Linked Data - The Story So Far. International Journal on Semantic Web and Information Systems **Special Issue on Linked Data** (2009)

24. Manola, F., Miller, E., "RDF Primer." McBride, B. (ed.): W3C Recommendation (2004)

25. Brickley, D., Guha, R.V., "RDF Schema." McBride, B. (ed.): W3C Recommendation (2004)

26. Hitzler, P., Krötzsch, M., Parsia, B., Patel-Schneider, P.F., Rudolph, S.: OWL 2 Web Ontology Language Primer. W3C (2009)

27. Hayes, P., "RDF Semantics," McBride, B. (ed.), W3C Recommendation (2004)

28. Patel-Schneider, P.F., Hayes, P., Horrocks, I.: OWL Web Ontology Language Semantics and Abstract Syntax W3C Recommendation (2004)

29. Protein knowledgebase: Uniprot, http://www.uniprot.org/, retrieved on July 15, 2010.

30. Wishart, D.S., Knox, C., Guo, A.C., Cheng, D., Shrivastava, S., Tzur, D., Gautam, B., Hassanali, M.: DrugBank: a knowledgebase for drugs, drug actions and drug targets. Nucleic Acids Res. **36** (2008) 901-906

31. BioPAX : Biological Pathways Exchange, , http://www.biopax.org/, retrieved on July 15, 2010.

32. Entrez, http://ww.ncbi.nlm.nih.gov/Entrez/, retrieved on July 15, 2010.

33. Unified Medical Language System (UMLS), http://www.nlm.nih.gov/research/umls/, retrieved on July 15, 2010.

34. Buneman, P., Khanna, S., Tan, W.C. : Data Provenance: Some Basic Issues. Lecture Notes in Computer Science **1974** (2000) 87--93

35. Cui, Y., Widom, J.: Practical Lineage Tracing in Data Warehouses. 16th ICDE. IEEE Computer Society, San Diego, California (2000)

36. Buneman, P., Khanna , S., Tan, W.C.: Why and Where: A Characterization of Data Provenance. 8th International Conference on Database Theory (2001) 316 - 330

37. Green, T.J., Karvounarakis, G. ,Tannen, V.: Provenance Semirings. ACMSIGMOD-SIGACTSIGART Symposium on Principles of database systems (PODS) (2007) 675–686

38. Simmhan, Y.L., Plale, A.B., Gannon, A. D.: A survey of data provenance in e-science SIGMOD Rec. **34** (2005) 31 - 36

39. IPAW 2008, www.sci.utah.edu/ipaw2008/, retrieved on July 15, 2010.

40. McGuinness, D.L., van Harmelen, F., "OWL Web Ontology Language Overview." W3C Recommendation (2004)

41. Schulze-Kremer, S.: Ontologies for molecular biology and bioinformatics. Silico Biol. **2** (2002) 179-193

42. Miles, S.: Electronically Querying for the Provenance of Entities. Third International Provenance and Annotation Workshop, Chicago, USA (2006) 184-192

43. Sahoo, S.S., Barga, R.S., Goldstein, J., Sheth, A. : Provenance Algebra and Materialized View-based Provenance Management. Microsoft Research Technical Report (2008)

44. Sahoo, S.S., Bodenreider, O., Hitzler, P., Sheth, A., Thirunarayan, K.: Provenance Context Entity (PaCE): Scalable Provenance Tracking for Scientific RDF Data. Proceedings of the 22nd International Conference on Scientific and Statistical Database Management SSDBM2010 Heidelberg, Germany, (2010) 461-470

45. Sahoo, S.S., Thomas, C., Sheth, A., York, W. S., and Tartir, S.: Knowledge modeling and its application in life sciences: a tale of two ontologies. Proceedings of the 15th international Conference on World Wide Web WWW '06 Edinburgh, Scotland (2006) 317-326

46. Sahoo, S.S., Weatherly, D.B., Muttharaju, R., Anantharam, P., Sheth, A., Tarleton, R.L.: Ontology-driven Provenance Management in eScience: An Application in Parasite Research. In: R. Meersman, T.D.e.a. (ed.): The 8th International Conference on Ontologies, DataBases, and Applications of Semantics (ODBASE 09). Springer Verlag, Vilamoura, Algarve-Portugal (2009) 992-1009

47. Sahoo, S.S., Zeng, K., Bodenreider, O., Sheth, A.P.: From "glycosyltransferase" to "congenital muscular dystrophy": Integrating knowledge from NCBI Entrez Gene and the Gene Ontology. In: (Eds), K.K.e.a. (ed.): Medinfo. IOS Press, Brisbane Australia (2007) 1260-1264.

48. Sahoo, S.S., Bodenreider, O., Rutter, J.L.,Skinner, K.J., Sheth, A.P.: An ontology-driven semantic mashup of gene and biological pathway information: Application to the domain of nicotine dependence. Journal of Biomedical Informatics **41** (2008) 752-765

49. Sahoo, S.S., Thomas, C., Sheth, A., Henson, C., York, W.S.: GLYDE-an expressive XML standard for the representation of glycan structure. Carbohydr Res. **340** (2005) 2802-2807

50. Missier, P., Sahoo, S.S., Zhao, J., Goble, C., Sheth, A.: Janus: from Workflows to Semantic Provenance and Linked Open Data. IPAW 2010, Troy, NY (2010)

51. Patni, H., Sahoo, S.S., Henson, C., Sheth, A.: Provenance Aware Linked Sensor Data. 2nd Workshop on Trust and Privacy on the Social and Semantic Web, Co-located with ESWC2010, Heraklion Greece (2010)

52. Sahoo, S.S., Sheth, A. : Provenir ontology: Towards a Framework for eScience Provenance Management. Microsoft eScience Workshop. Microsoft Research, Pittsburgh, USA (2009)

53. Sahoo, S.S., Bodenreider, O., Zeng, K., Sheth, A.P.: An Experiment in Integrationg Large Biomedical Knowledge Resources with RDF: Application to Associating Genotype and Phenotype Information. workshop on Health Care and Life Sciences Data Integration for the Semantic Web at the 16th International World Wide Web Conference (WWW2007), Banff, Canada (2007)

54. Sahoo, S.S., Sheth, A.P., York, W.S., Miller J.A.: Semantic Web Services for N-glycosylation Process. International Symposium on Web Services for Computational Biology and Bioinformatics, Blacksburg, VA, USA (2005)

55. Sahoo, S.S., Sheth, A.: Bioinformatics applications of Web Services, Web Processes and role of Semantics. In: Cardoso, J., Sheth, A. (ed.): Semantic Web Processes and Their Applications. Springer (2006) 305–322

56. Sahoo, S.S., Sheth, A.P., Hunter, B., York, W.S.: SemBOWSER - Semantic Biological Web Services Registry. In: Cheung, C.J.O.B.a.K.-H. (ed.): Semantic Web: Revolutionizing Knowledge Discovery in the Life Sciences. Springer (2007)

57. Pierce, M.J., (PI), Sheth, A., York, W.S., Miller, J.A., Kochut, K. (Co-PI): Integrated Technology Resource for Biomedical Glycomics. NIH (July 1, 2003 - June 30, 2008.)

58. Tan, W.C.: Provenance in Databases: Past, Current, and Future. IEEE Data Eng. Bull. **30** (2007) 3 -12

59. Goble, C.: Position Statement: Musings on Provenance, Workflow and (Semantic Web) Annotations for Bioinformatics. Workshop on Data Derivation and Provenance, Chicago (2002)

60. Wong, S.C., Miles, S., Fang, W., Groth, P. and Moreau, L. : Validation of E-Science Experiments using a Provenance-based Approach.: 4th UK e-Science All Hands Meeting (AHM), Nottingham, UK (2005)

61. Ontology Matching, http://www.ontologymatching.org/, retrieved on July 15, 2010.

62. McBride, B.: Jena: A Semantic Web Toolkit. IEEE Internet Computing **6** (Nov. 2002) 55-59

63. Sesame, http://openRDF.org, retrieved on July 15, 2010.

64. Haarslev, V., Möller, R.: Racer: A Core Inference Engine for the Semantic Web. 2nd International Workshop on Evaluation of Ontology-based Tools (EON2003), located at the 2nd International Semantic Web Conference ISWC 2003, Sanibel Island, Florida, USA (2003) 27–36

65. Sirin, E., Parsia, B., Cuenca Grau, B., Kalyanpur, A., Katz, Y.: Pellet: A practical OWL-DL reasoner. Journal of Web Semantics **2** (2007)

66. FaCT++, http://owl.man.ac.uk/factplusplus/, retrieved on July 15, 2010.

67. Geospatial ontology, http://www.w3.org/2005/Incubator/geo/XGR-geo-ont/, retrieved on July 15, 2010.

68. Environment ontology, http://sweet.jpl.nasa.gov/, retrieved on July 15, 2010.

69. Prud'hommeaux, E., Seaborne, A., "SPARQL Query Language for RDF", W3C Recommendation. W3C (2008)

70. Das, S.: RDF Support in Oracle RDBMS. Oracle Technical Presentation

71. Wood, D., Gearon, P., Adams, T.: Kowari: A platform for semantic web storage and analysis.: 14th International WWW Conference, Chiba Japan (2005)

72. Mapping SQL Data to RDF, http://virtuoso.openlinksw.com/wiki/main/Main/VOSRDF, retrieved on July 15, 2010

73. welkin, http://simile.mit.edu/welkin/, retrieved on July 15, 2010.

74. Noy, N.F., Sintek, M.,Decker, S.,Crubezy, M.,Fergerson, R.W.,Musen, M.A.: Creating Semantic Web Contents with Protege-2000. IEEE Intelligent Systems **16** 60-71

75. Semantic Analytics Visualization, http://lsdis.cs.uga.edu/projects/semvis/, retrieved on July 15, 2010.

76. Arnold, J., Schuttler, H.-B, Logan, D., Battogtokh, D., Griffith, J., Arpinar, B., Bhandarkar, S., Datta, S., Kochut, K.J., Kraemer, E., Miller, J.A., Sheth, A., Strobel, G., Taha, T., Aleman-Meza, B., Doss, J., Harris, L., Nyong, A.: Metabolomics. Handbook of Industrial Mycology. Marcel Dekker, NY (2004) 597-633

77. Zhao, J., Goble, C., and Stevens, R.: Semantic web applications to e-science in silico experiments.: Proceedings of the 13th international World Wide Web Conference on Alternate Track Papers &Amp; Posters ACM Press, New York, NY, New York, NY, USA (2004) 284-285

78. Zhao, J., Wroe, C., Goble, C., Stevens, R., Quan, D., Greenwood, M.: Using Semantic Web Technologies for Representing e-Science Provenance. 3rd International Semantic Web Conference ISWC2004, Vol. LNCS 3298. Springer, Hiroshima, Japan (2004)

79. Taylor, C.F., Paton, N.W., Garwood, K.L., Kirby, P.D., Stead, D.A., Yin, Z., Deutsch, E.W., Selway, L., Walker, J., Riba-Garcia, I., Mohammed, S., Deery, M.J., Howard , Dunkley, T., Aebersold, R., Kell, D.B., Lilley, K.S., Roepstorff, P., Yates, J.R. 3rd, Brass, A., Brown, A.J., Cash, P., Gaskell, S.J., Hubbard, S.J., Oliver, S.G.: A systematic approach to modeling, capturing, and disseminating proteomics experimental data. Nat. Biotechnol. **21** (2003) 247-254

80. The Ontology for Biomedical Investigations, http://obi-ontology.org, retrieved on July 15, 2010.

81. Oberle, D., Ankolekar, A., Hitzler, P., Cimiano, P., Schmidt, C., Weiten, M., Loos, B., Porzel, R., Zorn, H.-P., Micelli, V., Sintek, M., Kiesel, M., Mougouie, B., Vembu, S., Baumann, S., Romanelli, M., Buitelaar, P., Engel, R., Sonntag, D., Reithinger, N., Burkhardt, F., Zhou, J.: DOLCE ergo SUMO: On Foundational and Domain Models in SWIntO (SmartWeb Integrated Ontology). Journal of Web Semantics: Science, Services and Agents on the World Wide Web (2007)

82. Niles, I., Pease, A. : Towards a Standard Upper Ontology. In: Welty, C., Smith, B. (ed.): 2nd International Conference on Formal Ontology in Information Systems (FOIS-2001), Ogunquit, Maine (2001)

83. Smith, B., Ceusters, W., Klagges, B., Kohler, J., Kumar, A., Lomax, J., Mungall, C., Neuhaus, F., Rector, A.L., Rosse, C.: Relations in biomedical ontologies. Genome Biol **6** (2005) R46

84. Goble, C., Wolstencroft, K., Goderis, A., Hull, D., Zhao, J., Alper, P., Lord, P., Wroe, C., Belhajjame, K., Turi, D., Stevens, R., Oinn, T., De Roure, D.: Knowledge discovery for biology with Taverna: Producing and consuming semantics in the Web of Science. In: Baker, C.J.O., Cheung, K.-H. (eds.): Semantic Web: Revolutionizing knowledge discovery in the life sciences. Springer, New York (2007) 355-395

85. Kanehisa, M., Goto, S., Kawashima, S., Okuno, Y., and Hattori, M.: The KEGG resources for deciphering the genome. Nucleic Acids Res. **32** (2004) D277-D280

86. Romero, P., Wagg, J., Green, M.L., Kaiser, D., Krummenacker, M., and P.D. Karp: Computational prediction of human metabolic pathways from the complete human genome. Genome Biology **6** (2004) 1-17

87. Simmhan, Y.L., Plale, B., Gannon, D.: Karma2: Provenance management for data driven workflows. International Journal of Web Services Research **5** (2008)

88. Open Provenance Model, http://openprovenance.org/, retrieved on July 15, 2010.

89. Hartig, O., Zhao, J.: Publishing and Consuming Provenance Metadata on the Web of Linked Data. 3rd International Provenance and Annotation Workshop (IPAW), Troy, New York, USA (2010)

90. McGuinness, D.L., Pinheiro da Silva, P.: Explaining Answers from the Semantic Web: The Inference Web Approach. Journal of Web Semantics **1** (2004) 397-413

91. Chiticariu, L., Tan, W., Vijayvargiya, G.: DBNotes: a post-it system for relational databases based on provenance.: ACM SIGMOD international Conference on Management of Data. ACM, New York, NY, Baltimore, Maryland (2005) 942-944

92. Widom, J.: Trio: A System for Data, Uncertainty, and Lineage. In: Aggarwal, C. (ed.): Managing and Mining Uncertain Data. Springer (2008)

93. Borgida, A.: Description Logics in Data Management. IEEE Transactions on Knowledge and Data Engineering **7** (1995) 671 - 682

94. Moreau, L., Clifford, B., Freire, J., Gil, Y., Groth, P., Futrelle, J., Kwasnikowska, N., Miles, S., Missier, P., Myers, J., Simmhan, Y., Stephan, E. and Van den Bussche, J.: The Open Provenance Model --- Core Specification (v1.1). (2009)

95. Provenance Challenge, http://twiki.ipaw.info/, retrieved on July 15, 2010

96. Widom, J.: Trio: A System for Integrated Management of Data, Accuracy, and Lineage. Second Biennial Conference on Innovative Data Systems Research (CIDR '05), Pacific Grove, California (2005)

97. Karvounarakis, G., Ives, Z. G., Tannen, V.: Querying Data Provenance Proceedings of the 2010 international Conference on Management of Data. ACM, New York, NY, Indianapolis, Indiana, USA (2010) 951-962

98. Anand, M.K., Bowers, S., Ludäscher, B.: Techniques for efficiently querying scientific workflow provenance graphs. . In: I. Manolescu, S.S., J. Teubner, M. Kitsuregawa, A. Leger, F. Naumann, A. Ailamaki, F. Ozcan (ed.): Proceedings of the 13th international Conference on Extending Database Technology, Vol. 426. ACM, New York, NY, Lausanne, Switzerland (2010) 287-298

99. Geerts, F., Kementsietsidis, A., Milano, D.: MONDRIAN: Annotating and querying databases through colors and blocks. International Conference on Data Engineering (ICDE), Atlanta, GA (2006)

100. Kementsietsidis, A., Wang, M.: Provenance query evaluation: what's so special about it? : Proceeding of the 18th ACM Conference on information and Knowledge Management. ACM, New York, NY, Hong Kong, China (2009) 681-690

101. Chapman, A.P., Jagadish, H. V., Ramanan, P.: Efficient provenance storage. ACM SIGMOD international Conference on Management of Data. ACM, New York, NY, Vancouver, Canada (2008) 993-1006

102. Davidson, S.B., Freire, J.: Provenance and scientific workflows: challenges and opportunities. ACM SIGMOD international conference on Management of data. ACM New York, NY, USA Vancouver, Canada (2008)

103. Simmhan, Y., Plale, B., Gannon, D., Marru, S.: Performance Evaluation of the Karma Provenance Framework for Scientific Workflows. In: Moreau, L., Foster, I. (ed.): Provenance and Annotation of Data. Springer Berlin / Heidelberg (2006) 222-236

104. Project Neptune, http://www.neptune.washington.edu/, retrieved on July 15, 2010.

105. Semantics and Services enabled Problem Solving Environment for Tcruzi, http://wiki.knoesis.org/index.php/Trykipedia, retrieved on July 15, 2010.

106. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F (ed.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press (2003)

107. Klyne, G., Carroll, J. J.: Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation (2004)

108. Wang, Y.R., Madnick, S. E. : A Polygen Model for Heterogeneous Database Systems: The Source Tagging Perspective. 16th VLDB Conference (1990) 519–538

109. Lee, T., Bressan, S: Multimodal Integration of Disparate Information Sources with Attribution. Entity Relationship Workshop on Information Retrieval and Conceptual Modeling (1997)

110. Alexe, B., Chiticariu, L., Tan. W.-C.: SPIDER: a Schema mapPIng DEbuggeR. . VLDB (2006) 1179–1182

111. Haas, L.M., Hern´andez, M. A. , Ho, H. , Popa, L., Roth, M.: Clio Grows Up: From Research Prototype to Industrial Tool. ACM SIGMOD, Baltimore, MD (2005) 805–810

112. Holland, D.A., Seltzer, M.I., Braun, U., Muniswamy-Reddy, K.: PASSing the provenance challenge. Concurrency and Control: Practice and Experience **20** (2008) 531-540

113. Zhao, J., Goble, C., Stevens, R., Turi, D. : Mining taverna's semantic web of provenance. Journal of Concurrency and Computation:Practice and Experience (2007)

114. Chong, E.I., Das, S., Eadon, G., and Srinivasan, J.: An efficient SQL-based RDF querying scheme.: 31st international Conference on Very Large Data Bases. VLDB Endowment, Trondheim, Norway (2005) 1216-1227

115. Angles, R., Gutierrez, C.: Survey of graph database models. ACM Comput. Surv. **40** (2008) 1-39

116. Vardi, M.: The Complexity of Relational Query Languages. 14th Ann. ACM Symp. Theory of Computing (STOC ’82) (1982) 137-146

117. Pérez, J., Arenas, M., Gutiérrez, C. : Semantics and Complexity of SPARQL. Int'l Semantic Web Conf. (ISWC '06), Vol. 4273/2006, Athens, GA (2006) 30-43

118. Trident Workflow Workbench, http://www.microsoft.com/mscorp/tc/trident.mspx, retrieved on July 15, 2010.

119. The VisTrails Project, http://www.vistrails.org/, retrieved on July 15, 2010.

120. Scheidegger, C., Koop, D., Santos, E., Vo, H., Callahan, S., Freire, J., Silva, C.: Tackling the Provenance Challenge One Layer at a Time. Concurrency and Computation:Practice and Experience **20** (2007) 473-483

121. Bodenreider, O.: The Unified Medical Language System (UMLS): integrating biomedical terminology. Nucleic Acids Res. **32** (2004) 267-270

122. Bodenreider, O., Rindflesch, T.C.: Advanced library services: Developing a biomedical knowledge repository to support advanced information management applications. Lister Hill National Center for Biomedical Communications, National Library of Medicine, Bethesda, Maryland (2006)

123. RDB2RDF XG Final Report. World Wide Web Consortium (2009).

124. MedlinePlus. Vol. 2010. National Library of Medicine (US), Bethesda (MD)

125. ter Horst, H.J.: Completeness, decidability and complexity of entailment for RDF Schema and a semantic extension involving the OWL vocabulary. Journal of Web Semantics **3** (2005) 79-115

126. Carroll, J.J., Bizer, C., Hayes, P., Stickler, P.: Named graphs, provenance and trust. 14th international Conference on World Wide Web. ACM, New York, NY, Chiba, Japan (2005) 613-622

127. Ding, L., Finin, T., Joshi, A., Peng, J., Pinheiro da Silva, P., McGuinness, D.L.: Tracking RDF Graph Provenance using RDF Molecules. Proceedings of the 4th International Semantic Web Conference (2005)

128. Flouris, G., Fundulaki, I., Pediaditis, P., Theoharis, Y., Christophides, V.: Coloring RDF Triples to Capture Provenance. International Semantic Web Conference 2009, Washington D.C., USA (2009) 196-212

129. Guha, R.V.: Contexts: A Formalization and Some Applications PhD Thesis, Stanford University (1991)

130. Guha, R.V., McCarthy, J.: Varieties of Contexts. CONTEXT 2003 (2003) 164–177

131. McCarthy, J.: Generality in artificial intelligence. Formalizing Common Sense: Papers by John McCarthy (1990) 226–236

132. Nayak, P.P.: Representing multiple theories. In: B. Hayes-Roth, R.K. (ed.): AAAI-94, Menlo Park, CA, USA (1994) 1154–1160

133. Buvaˇc, S., Mason, I.: Propositional logic of context. AAAI (1993) 412–419

134. Giunchiglia, F., Ghidini, C.: Local models semantics, or contextual reasoning = locality+compatibility. In: Anthony G. Cohn, L.S., Stuart C. Shapiro (ed.): KR'98: Principles of Knowledge Representation and Reasoning. Morgan Kaufmann, San Francisco, CA, USA (1998) 282-289

135. Guha, R., McCool, R., Fikes, R.: Contexts for the Semantic Web In: Sheila A. McIlraith, D.P., Frank van Harmelen (ed.): International Semantic Web Conference, Vol. 3298. Springer, Hiroshima, Japan (2004) 32-46

136. Ayers, A., Völkel, M.: Cool URIs for the Semantic Web. In: Sauermann, L., Cyganiak, R. (ed.): Woking Draft. W3C (2008)

137. Wikipedia, http://www.wikipedia.org/, retrieved on July 15, 2010.

138. Clinical Trials, http://clinicaltrials.gov/, retrieved on July 15, 2010.

139. Weatherly, B., Atwood, J., Minning, T., Cavola, C., Tarleton, R., Orlando, R.: A heuristic method for assigning a false-discovery rate for protein identifications from Mascot database search results. Mol. Cell. Proteomics **4** (2005) 762-772

140. Kelly, B.K., Anderson, P. E., Reo, N. V., DelRaso, N. J. , Doom, T. E., Raymer, M. L.: A proposed statistical protocol for the analysis of metabolic toxicological data derived from NMR spectroscopy. 7th IEEE International Conference on Bioinformatics and Bioengineering (BIBE 2007), Vol. II, Cambridge - Boston, Massachusetts, USA (2007) 1414-1418

141. Oracle Life Sciences Users Group, http://www.olsug.org/, retrieved on July 15, 2010.

142. Aurrecoechea, C., M. Heiges, H. Wang, Z. Wang, S. Fischer, P. Rhodes, J. Miller, E. Kraemer, C. J. Stoeckert, Jr., D. S. Roos, and J. C. Kissinger: ApiDB: integrated resources for the apicomplexan bioinformatics resource center. Nucleic Acids Research **35** (2007) 427-430

143. NCI Thesaurus, http://ncit.nci.nih.gov, retrieved on July 15, 2010.

144. Nguyen, V., Sahoo, S.S., Parikh, P., Minning, T., Weatherly, B., Logan, F., Sheth, A., Tarleton, R., "Biomedical Ontologies for Parasite Research," ISMB2010, July 11-13, Boston, MA, USA.

145. Hobbs, J.R., Pan, F.: Time Ontology in OWL W3C Working Draft (2006)

146. Sahoo, S.S., Barga, R.S., Sheth, A.P., Thirunarayan, K., Hitzler, P.: PrOM: A Semantic Web Framework for Provenance Management in Science. Kno.e.sis Center, Wright State University (2009)

147. SPCDIS: Semantic Provenance Capture in Data Ingest Systems, http://tw.rpiscrews.us/wiki/SPCDIS, retrieved on July 15, 2010.

148. Inference Web, http://inference-web.org/, retrieved on July 15, 2010.

53. Sahoo, S.S., Bodenreider, O., Zeng, K., Sheth, A.P.: An Experiment in Integrationg Large Biomedical Knowledge Resources with RDF: Application to Associating Genotype and Phenotype Information. workshop on Health Care and Life Sciences Data Integration for the Semantic Web at the 16th International World Wide Web Conference (WWW2007), Banff, Canada (2007)

54. Sahoo, S.S., Sheth, A.P., York, W.S., Miller J.A.: Semantic Web Services for N-glycosylation Process. International Symposium on Web Services for Computational Biology and Bioinformatics, Blacksburg, VA, USA (2005)

55. Sahoo, S.S., Sheth, A.: Bioinformatics applications of Web Services, Web Processes and role of Semantics. In: Cardoso, J., Sheth, A. (ed.): Semantic Web Processes and Their Applications. Springer (2006) 305–322

56. Sahoo, S.S., Sheth, A.P., Hunter, B., York, W.S.: SemBOWSER - Semantic Biological Web Services Registry. In: Cheung, C.J.O.B.a.K.-H. (ed.): Semantic Web: Revolutionizing Knowledge Discovery in the Life Sciences. Springer (2007)

57. Pierce, M.J., (PI), Sheth, A., York, W.S., Miller, J.A., Kochut, K. (Co-PI): Integrated Technology Resource for Biomedical Glycomics. NIH (July 1, 2003 - June 30, 2008.)

58. Tan, W.C.: Provenance in Databases: Past, Current, and Future. IEEE Data Eng. Bull. **30** (2007) 3 -12

59. Goble, C.: Position Statement: Musings on Provenance, Workflow and (Semantic Web) Annotations for Bioinformatics. Workshop on Data Derivation and Provenance, Chicago (2002)

60. Wong, S.C., Miles, S., Fang, W., Groth, P. and Moreau, L. : Validation of E-Science Experiments using a Provenance-based Approach.: 4th UK e-Science All Hands Meeting (AHM), Nottingham, UK (2005)

61. Ontology Matching, http://www.ontologymatching.org/, retrieved on July 15, 2010.

62. McBride, B.: Jena: A Semantic Web Toolkit. IEEE Internet Computing **6** (Nov. 2002) 55-59

63. Sesame, http://openRDF.org, retrieved on July 15, 2010.

64. Haarslev, V., Möller, R.: Racer: A Core Inference Engine for the Semantic Web. 2nd International Workshop on Evaluation of Ontology-based Tools (EON2003), located at the 2nd International Semantic Web Conference ISWC 2003, Sanibel Island, Florida, USA (2003) 27–36

65. Sirin, E., Parsia, B., Cuenca Grau, B., Kalyanpur, A., Katz, Y.: Pellet: A practical OWL-DL reasoner. Journal of Web Semantics **2** (2007)

66. FaCT++, http://owl.man.ac.uk/factplusplus/, retrieved on July 15, 2010.

67. Geospatial ontology, http://www.w3.org/2005/Incubator/geo/XGR-geo-ont/, retrieved on July 15, 2010.

68. Environment ontology, http://sweet.jpl.nasa.gov/, retrieved on July 15, 2010.

69. Prud'hommeaux, E., Seaborne, A., "SPARQL Query Language for RDF", W3C Recommendation. W3C (2008)

70. Das, S.: RDF Support in Oracle RDBMS. Oracle Technical Presentation

71. Wood, D., Gearon, P., Adams, T.: Kowari: A platform for semantic web storage and analysis.: 14th International WWW Conference, Chiba Japan (2005)

72. Mapping SQL Data to RDF, http://virtuoso.openlinksw.com/wiki/main/Main/VOSRDF, retrieved on July 15, 2010

73. welkin, http://simile.mit.edu/welkin/, retrieved on July 15, 2010.

74. Noy, N.F., Sintek, M.,Decker, S.,Crubezy, M.,Fergerson, R.W.,Musen, M.A.: Creating Semantic Web Contents with Protege-2000. IEEE Intelligent Systems **16** 60-71

75. Semantic Analytics Visualization, http://lsdis.cs.uga.edu/projects/semvis/, retrieved on July 15, 2010.

76. Arnold, J., Schuttler, H.-B, Logan, D., Battogtokh, D., Griffith, J., Arpinar, B., Bhandarkar, S., Datta, S., Kochut, K.J., Kraemer, E., Miller, J.A., Sheth, A., Strobel, G., Taha, T., Aleman-Meza, B., Doss, J., Harris, L., Nyong, A.: Metabolomics. Handbook of Industrial Mycology. Marcel Dekker, NY (2004) 597-633

77. Zhao, J., Goble, C., and Stevens, R.: Semantic web applications to e-science in silico experiments.: Proceedings of the 13th international World Wide Web Conference on Alternate Track Papers &Amp; Posters ACM Press, New York, NY, New York, NY, USA (2004) 284-285

78. Zhao, J., Wroe, C., Goble, C., Stevens, R., Quan, D., Greenwood, M.: Using Semantic Web Technologies for Representing e-Science Provenance. 3rd International Semantic Web Conference ISWC2004, Vol. LNCS 3298. Springer, Hiroshima, Japan (2004)

79. Taylor, C.F., Paton, N.W., Garwood, K.L., Kirby, P.D., Stead, D.A., Yin, Z., Deutsch, E.W., Selway, L., Walker, J., Riba-Garcia, I., Mohammed, S., Deery, M.J., Howard , Dunkley, T., Aebersold, R., Kell, D.B., Lilley, K.S., Roepstorff, P., Yates, J.R. 3rd, Brass, A., Brown, A.J., Cash, P., Gaskell, S.J., Hubbard, S.J., Oliver, S.G.: A systematic approach to modeling, capturing, and disseminating proteomics experimental data. Nat. Biotechnol. **21** (2003) 247-254

80. The Ontology for Biomedical Investigations, http://obi-ontology.org, retrieved on July 15, 2010.

81. Oberle, D., Ankolekar, A., Hitzler, P., Cimiano, P., Schmidt, C., Weiten, M., Loos, B., Porzel, R., Zorn, H.-P., Micelli, V., Sintek, M., Kiesel, M., Mougouie, B., Vembu, S., Baumann, S., Romanelli, M., Buitelaar, P., Engel, R., Sonntag, D., Reithinger, N., Burkhardt, F., Zhou, J.: DOLCE ergo SUMO: On Foundational and Domain Models in SWIntO (SmartWeb Integrated Ontology). Journal of Web Semantics: Science, Services and Agents on the World Wide Web (2007)

82. Niles, I., Pease, A. : Towards a Standard Upper Ontology. In: Welty, C., Smith, B. (ed.): 2nd International Conference on Formal Ontology in Information Systems (FOIS-2001), Ogunquit, Maine (2001)

83. Smith, B., Ceusters, W., Klagges, B., Kohler, J., Kumar, A., Lomax, J., Mungall, C., Neuhaus, F., Rector, A.L., Rosse, C.: Relations in biomedical ontologies. Genome Biol **6** (2005) R46

84. Goble, C., Wolstencroft, K., Goderis, A., Hull, D., Zhao, J., Alper, P., Lord, P., Wroe, C., Belhajjame, K., Turi, D., Stevens, R., Oinn, T., De Roure, D.: Knowledge discovery for biology with Taverna: Producing and consuming semantics in the Web of Science. In: Baker, C.J.O., Cheung, K.-H. (eds.): Semantic Web: Revolutionizing knowledge discovery in the life sciences. Springer, New York (2007) 355-395

85. Kanehisa, M., Goto, S., Kawashima, S., Okuno, Y., and Hattori, M.: The KEGG resources for deciphering the genome. Nucleic Acids Res. **32** (2004) D277-D280

86. Romero, P., Wagg, J., Green, M.L., Kaiser, D., Krummenacker, M., and P.D. Karp: Computational prediction of human metabolic pathways from the complete human genome. Genome Biology **6** (2004) 1-17

87. Simmhan, Y.L., Plale, B., Gannon, D.: Karma2: Provenance management for data driven workflows. International Journal of Web Services Research **5** (2008)

88. Open Provenance Model, http://openprovenance.org/, retrieved on July 15, 2010.

89. Hartig, O., Zhao, J.: Publishing and Consuming Provenance Metadata on the Web of Linked Data. 3rd International Provenance and Annotation Workshop (IPAW), Troy, New York, USA (2010)

90. McGuinness, D.L., Pinheiro da Silva, P.: Explaining Answers from the Semantic Web: The Inference Web Approach. Journal of Web Semantics **1** (2004) 397-413

91. Chiticariu, L., Tan, W., Vijayvargiya, G.: DBNotes: a post-it system for relational databases based on provenance.: ACM SIGMOD international Conference on Management of Data. ACM, New York, NY, Baltimore, Maryland (2005) 942-944

92. Widom, J.: Trio: A System for Data, Uncertainty, and Lineage. In: Aggarwal, C. (ed.): Managing and Mining Uncertain Data. Springer (2008)

93. Borgida, A.: Description Logics in Data Management. IEEE Transactions on Knowledge and Data Engineering **7** (1995) 671 - 682

94. Moreau, L., Clifford, B., Freire, J., Gil, Y., Groth, P., Futrelle, J., Kwasnikowska, N., Miles, S., Missier, P., Myers, J., Simmhan, Y., Stephan, E. and Van den Bussche, J.: The Open Provenance Model --- Core Specification (v1.1). (2009)

95. Provenance Challenge, http://twiki.ipaw.info/, retrieved on July 15, 2010

96. Widom, J.: Trio: A System for Integrated Management of Data, Accuracy, and Lineage. Second Biennial Conference on Innovative Data Systems Research (CIDR '05), Pacific Grove, California (2005)

97. Karvounarakis, G., Ives, Z. G., Tannen, V.: Querying Data Provenance Proceedings of the 2010 international Conference on Management of Data. ACM, New York, NY, Indianapolis, Indiana, USA (2010) 951-962

98. Anand, M.K., Bowers, S., Ludäscher, B.: Techniques for efficiently querying scientific workflow provenance graphs. . In: I. Manolescu, S.S., J. Teubner, M. Kitsuregawa, A. Leger, F. Naumann, A. Ailamaki, F. Ozcan (ed.): Proceedings of the 13th international Conference on Extending Database Technology, Vol. 426. ACM, New York, NY, Lausanne, Switzerland (2010) 287-298

99. Geerts, F., Kementsietsidis, A., Milano, D.: MONDRIAN: Annotating and querying databases through colors and blocks. International Conference on Data Engineering (ICDE), Atlanta, GA (2006)

100. Kementsietsidis, A., Wang, M.: Provenance query evaluation: what's so special about it? : Proceeding of the 18th ACM Conference on information and Knowledge Management. ACM, New York, NY, Hong Kong, China (2009) 681-690

101. Chapman, A.P., Jagadish, H. V., Ramanan, P.: Efficient provenance storage. ACM SIGMOD international Conference on Management of Data. ACM, New York, NY, Vancouver, Canada (2008) 993-1006

102. Davidson, S.B., Freire, J.: Provenance and scientific workflows: challenges and opportunities. ACM SIGMOD international conference on Management of data. ACM New York, NY, USA Vancouver, Canada (2008)

103. Simmhan, Y., Plale, B., Gannon, D., Marru, S.: Performance Evaluation of the Karma Provenance Framework for Scientific Workflows. In: Moreau, L., Foster, I. (ed.): Provenance and Annotation of Data. Springer Berlin / Heidelberg (2006) 222-236

104. Project Neptune, http://www.neptune.washington.edu/, retrieved on July 15, 2010.

105. Semantics and Services enabled Problem Solving Environment for Tcruzi, http://wiki.knoesis.org/index.php/Trykipedia, retrieved on July 15, 2010.

106. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F (ed.): The Description Logic Handbook: Theory, Implementation, and Applications. Cambridge University Press (2003)

107. Klyne, G., Carroll, J. J.: Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation (2004)

108. Wang, Y.R., Madnick, S. E. : A Polygen Model for Heterogeneous Database Systems: The Source Tagging Perspective. 16th VLDB Conference (1990) 519–538

109. Lee, T., Bressan, S: Multimodal Integration of Disparate Information Sources with Attribution. Entity Relationship Workshop on Information Retrieval and Conceptual Modeling (1997)

110. Alexe, B., Chiticariu, L., Tan. W.-C.: SPIDER: a Schema mapPIng DEbuggeR. . VLDB (2006) 1179–1182

111. Haas, L.M., Hern´andez, M. A. , Ho, H. , Popa, L., Roth, M.: Clio Grows Up: From Research Prototype to Industrial Tool. ACM SIGMOD, Baltimore, MD (2005) 805–810

112. Holland, D.A., Seltzer, M.I., Braun, U., Muniswamy-Reddy, K.: PASSing the provenance challenge. Concurrency and Control: Practice and Experience **20** (2008) 531-540

113. Zhao, J., Goble, C., Stevens, R., Turi, D. : Mining taverna's semantic web of provenance. Journal of Concurrency and Computation:Practice and Experience (2007)

114. Chong, E.I., Das, S., Eadon, G., and Srinivasan, J.: An efficient SQL-based RDF querying scheme.: 31st international Conference on Very Large Data Bases. VLDB Endowment, Trondheim, Norway (2005) 1216-1227

115. Angles, R., Gutierrez, C.: Survey of graph database models. ACM Comput. Surv. **40** (2008) 1-39

116. Vardi, M.: The Complexity of Relational Query Languages. 14th Ann. ACM Symp. Theory of Computing (STOC ’82) (1982) 137-146

117. Pérez, J., Arenas, M., Gutiérrez, C. : Semantics and Complexity of SPARQL. Int'l Semantic Web Conf. (ISWC '06), Vol. 4273/2006, Athens, GA (2006) 30-43

118. Trident Workflow Workbench, http://www.microsoft.com/mscorp/tc/trident.mspx, retrieved on July 15, 2010.

119. The VisTrails Project, http://www.vistrails.org/, retrieved on July 15, 2010.

120. Scheidegger, C., Koop, D., Santos, E., Vo, H., Callahan, S., Freire, J., Silva, C.: Tackling the Provenance Challenge One Layer at a Time. Concurrency and Computation:Practice and Experience **20** (2007) 473-483

121. Bodenreider, O.: The Unified Medical Language System (UMLS): integrating biomedical terminology. Nucleic Acids Res. **32** (2004) 267-270

122. Bodenreider, O., Rindflesch, T.C.: Advanced library services: Developing a biomedical knowledge repository to support advanced information management applications. Lister Hill National Center for Biomedical Communications, National Library of Medicine, Bethesda, Maryland (2006)

123. RDB2RDF XG Final Report. World Wide Web Consortium (2009).

124. MedlinePlus. Vol. 2010. National Library of Medicine (US), Bethesda (MD)

125. ter Horst, H.J.: Completeness, decidability and complexity of entailment for RDF Schema and a semantic extension involving the OWL vocabulary. Journal of Web Semantics **3** (2005) 79-115

126. Carroll, J.J., Bizer, C., Hayes, P., Stickler, P.: Named graphs, provenance and trust. 14th international Conference on World Wide Web. ACM, New York, NY, Chiba, Japan (2005) 613-622

127. Ding, L., Finin, T., Joshi, A., Peng, J., Pinheiro da Silva, P., McGuinness, D.L.: Tracking RDF Graph Provenance using RDF Molecules. Proceedings of the 4th International Semantic Web Conference (2005)

128. Flouris, G., Fundulaki, I., Pediaditis, P., Theoharis, Y., Christophides, V.: Coloring RDF Triples to Capture Provenance. International Semantic Web Conference 2009, Washington D.C., USA (2009) 196-212

129. Guha, R.V.: Contexts: A Formalization and Some Applications PhD Thesis, Stanford University (1991)

130. Guha, R.V., McCarthy, J.: Varieties of Contexts. CONTEXT 2003 (2003) 164–177

131. McCarthy, J.: Generality in artificial intelligence. Formalizing Common Sense: Papers by John McCarthy (1990) 226–236

132. Nayak, P.P.: Representing multiple theories. In: B. Hayes-Roth, R.K. (ed.): AAAI-94, Menlo Park, CA, USA (1994) 1154–1160

133. Buvaˇc, S., Mason, I.: Propositional logic of context. AAAI (1993) 412–419

134. Giunchiglia, F., Ghidini, C.: Local models semantics, or contextual reasoning = locality+compatibility. In: Anthony G. Cohn, L.S., Stuart C. Shapiro (ed.): KR'98: Principles of Knowledge Representation and Reasoning. Morgan Kaufmann, San Francisco, CA, USA (1998) 282-289

135. Guha, R., McCool, R., Fikes, R.: Contexts for the Semantic Web In: Sheila A. McIlraith, D.P., Frank van Harmelen (ed.): International Semantic Web Conference, Vol. 3298. Springer, Hiroshima, Japan (2004) 32-46

136. Ayers, A., Völkel, M.: Cool URIs for the Semantic Web. In: Sauermann, L., Cyganiak, R. (ed.): Woking Draft. W3C (2008)

137. Wikipedia, http://www.wikipedia.org/, retrieved on July 15, 2010.

138. Clinical Trials, http://clinicaltrials.gov/, retrieved on July 15, 2010.

139. Weatherly, B., Atwood, J., Minning, T., Cavola, C., Tarleton, R., Orlando, R.: A heuristic method for assigning a false-discovery rate for protein identifications from Mascot database search results. Mol. Cell. Proteomics **4** (2005) 762-772

140. Kelly, B.K., Anderson, P. E., Reo, N. V., DelRaso, N. J. , Doom, T. E., Raymer, M. L.: A proposed statistical protocol for the analysis of metabolic toxicological data derived from NMR spectroscopy. 7th IEEE International Conference on Bioinformatics and Bioengineering (BIBE 2007), Vol. II, Cambridge - Boston, Massachusetts, USA (2007) 1414-1418

141. Oracle Life Sciences Users Group, http://www.olsug.org/, retrieved on July 15, 2010.

142. Aurrecoechea, C., M. Heiges, H. Wang, Z. Wang, S. Fischer, P. Rhodes, J. Miller, E. Kraemer, C. J. Stoeckert, Jr., D. S. Roos, and J. C. Kissinger: ApiDB: integrated resources for the apicomplexan bioinformatics resource center. Nucleic Acids Research **35** (2007) 427-430

143. NCI Thesaurus, http://ncit.nci.nih.gov, retrieved on July 15, 2010.

144. Nguyen, V., Sahoo, S.S., Parikh, P., Minning, T., Weatherly, B., Logan, F., Sheth, A., Tarleton, R., "Biomedical Ontologies for Parasite Research," ISMB2010, July 11-13, Boston, MA, USA.

145. Hobbs, J.R., Pan, F.: Time Ontology in OWL W3C Working Draft (2006)

146. Sahoo, S.S., Barga, R.S., Sheth, A.P., Thirunarayan, K., Hitzler, P.: PrOM: A Semantic Web Framework for Provenance Management in Science. Kno.e.sis Center, Wright State University (2009)

147. SPCDIS: Semantic Provenance Capture in Data Ingest Systems, http://tw.rpiscrews.us/wiki/SPCDIS, retrieved on July 15, 2010.

148. Inference Web, http://inference-web.org/, retrieved on July 15, 2010.